EMPIRICAL ANALYSIS OF OPTICAL ATTENUATOR PERFORMANCE IN QUANTUM KEY DISTRIBUTION SYSTEMS USING A PARTICLE MODEL

THESIS

Thomas C. Adams, Captain, USAF

AFIT/GCS/ENG/12-01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GCS/ENG/12-01

EMPIRICAL ANALYSIS OF OPTICAL ATTENUATOR PERFORMANCE IN QUANTUM KEY DISTRIBUTION SYSTEMS USING A PARTICLE MODEL

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science

Thomas C. Adams, AS, BS

Captain, USAF

March 2012

AFIT/GCS/ENG/12-01

EMPIRICAL ANALYSIS OF OPTICAL ATTENUATOR PERFORMANCE IN QUANTUM KEY DISTRIBUTION SYSTEMS USING A PARTICLE MODEL

Thomas C. Adams, AS, BS
Captain, USAF

Approved:

_____	_____
Lt Col Jeffrey W. Humphries, PhD (Chairman)	Date
_____	_____
Michael R. Grimaila, PhD, CISM, CISSP (Member)	Date
_____	_____
Gerald Baumgartner, PhD, (Member)	Date

Abstract

The process for developing software has evolved since the dawn of computers with several paradigms for creating the code that runs on them. One area of software development focuses on the modeling of physical objects and processes to predict their behavior using simulation. An accurate model can provide system developers several advantages in their development efforts. Quantum key distribution networks currently represent an active area of development. Concerns over the ability of today's technology to secure communications in the future are driving quantum key distribution system development. Those systems possess several components that on the surface are logically easy to simulate using computers, but become more challenging as the complexity of actual implementation specifics are considered. Two components common to most quantum key distribution implementations are the signal source and an optical attenuator. Their role in the system is to provide the single photon per bit necessary to maintain theoretically perfect secrecy. How the photon pulse is modeled has a significant impact on the accuracy and performance of quantum channel components like the optical attenuator. Classical physics describe light using Maxwell's wave equations for electromagnetism. Quantum physics has demonstrated light also behaves as discrete particles referred to as photons. This paper looks at developing and characterizing the accuracy of software models for an optical attenuator as might be used in a quantum key distribution system using a particle-only model of the photon pulse.

Table of Contents

Page

List of Figures

List of Tables

List of Equations

EMPIRICAL ANALYSIS OF OPTICAL ATTENUATOR PERFORMANCE IN QUANTUM KEY DISTRIBUTION SYSTEMS USING A PARTICLE MODEL

I. Introduction

When implemented correctly, use of the one-time pad (OTP) method of encryption has been proven impossible to crack with all possible decryptions being equally likely [7]. With this provably perfect security, it would seem this system of encryption would dominate all others, but security comes at a price. One of the logistical issues associated with OTP use involves sharing, or distributing, a common key between the parties wishing to communicate without that key becoming available to any other party. This can be quite daunting, especially if the parties are separated and their method of communication is limited to means subject to interception and interrogation by a third party. One possible solution to this issue involves sharing the key in a manner that permits determining whether or not the key was intercepted during transmission. To that end, developers are looking at using elementary particles such as photons to carry the key and exploit the laws of quantum physics in the process of quantum key distribution (QKD).

With the early rudiments proposed in the 1970s [8] and implemented in 1991 [9], QKD relies on currently understood quantum mechanical principles to permit detecting the presence of an intercepting intermediary through the changes they induce when measuring the transmitted elementary particles. Although frequently described as waves, light has been shown to exhibit particle-like behavior and these particles have come to be known as photons.

While an ideal QKD system can demonstrate perfect secrecy, the difficulty in working with individual photons makes current implementations less than perfect. To date, there still exist no reliable single photon sources or detectors. A common means of approximating single photons involves using lasers with very brief pulse durations and intensities. Even still, the number of photons emitted is significantly above the single photon used to carry QKD information. Therefore, optical attenuators are used to decrease the laser's signal down to the useful level.

System developers are faced with making these and other compromises to deal with the non-idealities of actual QKD implementations resulting in deviations that are often exploitable. Modeling and simulation offer these system developers a means of predicting system behavior resulting from design decisions prior to physical implementation.

The Department of Defense considers modeling and simulation a "key enabler of DoD activities" and a "tool for achieving DoD goals" [10]. While a number of hardware implementations of QKD systems have been developed and tested, a software model for characterizing and simulating QKD systems could facilitate the rapid testing of changes to system parameters and components in order to predict system behavior. Software modeling of physical systems and their processes often involves tradeoffs between performance and fidelity.

The objective of this research is to develop software models for the initial components of a QKD quantum channel and characterize their accuracy through simulation. The research focuses specifically on the signal source as a group of particles or photon pulse, and an optical attenuator that reduces that photon pulse down to,

optimally, a single photon for carrying the information used by a BB84 QKD implementation over single-mode optical fiber.

While an optical attenuator's overall function is somewhat trivial, it reduces the intensity of signals passing through it by some amount, the means by which it performs its purpose can be quite complex when looked at the quantum level as in QKD.

If the quantum signal used were represented by a wave model, the optical attenuator would become a logically simple transformation with all the functionality rolled into a mathematical function that merely decreases the amplitude of the wave by some prescribed amount. At the macro level, this model is highly efficient requiring very little processing by the attenuator on the optical signal. Components downstream of the optical attenuator would perform their own transformations on the resulting quantum signal to ultimately convert the amplitude into a photon count, likely rounding down or up partial photons which can't exist and represent some amount of error in the transformations from wave to discrete particles.

Real optical attenuators are not mathematical functions. For this research effort, an attenuator utilizing geometrical misalignments, specifically lateral offset misalignments, is considered. While a highly efficient attenuator can be produced using a wave model for the quantum signal, this author proposes an attenuator using a particle model for the quantum signal is more realistic by simulating what occurs in the attenuator at its optical junction and therefore, is more accurate for simulating QKD implementations.

To accomplish these goals, the author creates models for a QKD photon pulse using a three dimensional matrix to represent the pulse in time and space, an adjustable

lateral offset attenuator used to restrict the passage of photons through it based on the locations of the photons relative to the attenuator's lateral offset settings, and supporting code to confirm and collect results on the accuracy and performance of the interactions between models to compare against the mathematically calculated expected values.

The research looks to answer the following questions:

- what does a QKD photon pulse look like spatially so as to model it as a group of particles,
- how should particles be distributed throughout the photon pulse,
- what dimensions should such a particle model possess,
- how should a lateral offset attenuator work given such a photon pulse,
- what limitations exist when simulating with these models?

One design goal of the model is to promote the rapid replacement of software functionality to allow numerous "what if" scenarios to allow a broad range of model fidelities available for rapid substitution to focus on any particular aspect or interaction of aspects within the QKD system.

Before modeling QKD, in whole or in part, it would be beneficial to define what constitutes a QKD system, some of its integral parts, and the physics behind their functionality.

II. Background

2.1 Overview

This chapter contains the principle ideas, equations, and insight used in pursuing this paper's research. The dichotomy between representing light as waves and particles,

central to quantum theory, is reviewed and serves as the basis behind the fundamental tenants of QKD. After a brief highlight of the basics behind QKD, two key components, optical fibers and optical attenuators, are looked at in detail to build the foundation for modeling them as software components.

2.2 Light As Waves

The Dutch physicist Christian Huygens was the first person to propose a "convincing" theory that light behaves as a wave in 1678 [1]. He described light as a plane wave whose points at any given point in time act as sources whose propagation as individual spherical waves combine to form a wavefront tangent to the spherical waves. Using this model, he successfully derived the laws of reflection and refraction [11]. Figure 1 below illustrates his model:

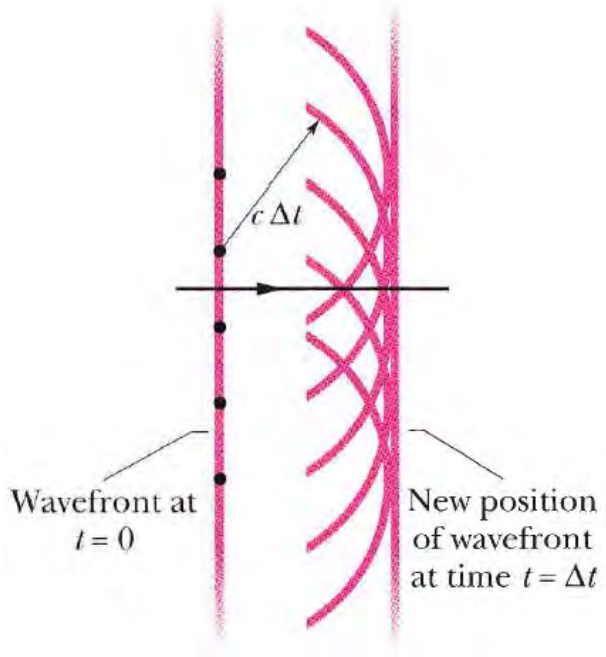

Figure 1: Huygen's Model For Light Propagation [1]

In Figure 1, arbitrary points selected from the plane on the left define the center of spheres that propagate at the speed of light c for some time Δt whose tangents define the location of the plane on the right.

Thomas Young's slit experiment in 1801 provided more concrete evidence to this wave model. Figure 2 below illustrates his experiment:

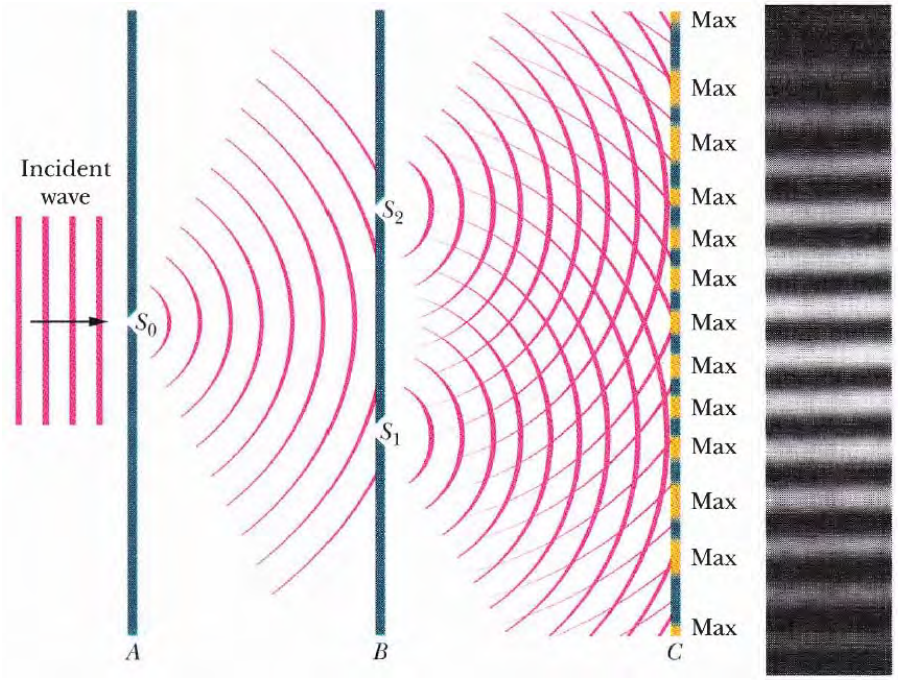

Figure 2: Thomas Young's Slit Experiment [1]

Figure 2 shows the setup and results of Thomas Young's experiment. Narrow slit S_0 in screen *A* allows light to impinge on two narrow slits S_1 and S_2 on screen *B*. As the light propagates towards screen *C*, the individual light sources created by slits S_1 and S_2 interfere with one another as expected if the light propagated as waves. The interference results in a pattern of distinct maxima and minima as shown on the right of Figure 2.

James Clerk Maxwell would come to define the characteristics of light, and electromagnetic energy in general, with equations bearing his name. But even these failed to explain the processes of emission and absorption where the theories of quantum mechanics are required [11] along with the concept of light as discrete particles.

2.3 Light As Particles

One of the first observations that light exists as particles came from the photoelectric effect. First observed by Heinrich Hertz in 1887, the photoelectric affect went unexplained for 18 years given only Maxwell's wave equations for light. Figure 3 below illustrates a setup for demonstrating the photoelectric effect:

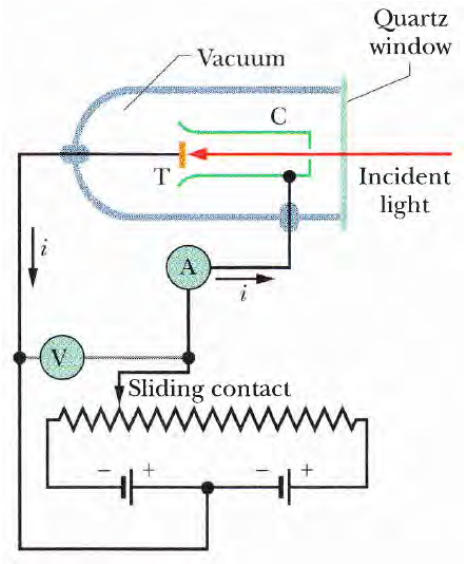

Figure 3: Apparatus For Demonstrating Photoelectric Effect [1]

In Figure 3 above, an incident beam of light is directed onto a metal surface *T* causing electrons to be emitted when the light's wavelength is short enough. By adjusting the sliding contact to vary resistance, a slightly negative potential measured by voltmeter

V across collector *C* and the metal surface *T* can be maintained such that electrons emitted from surface *T* are just barely repelled by collector *C*. The resulting current *i* is measured by ammeter *A* [1].

Tests show setting the potential measured at *V* such that the current read at *A* is just zero measures the maximum kinetic energy of an emitted electron as given in Equation 1 below:

$$K_{max} = eV_{stop} \tag{1}$$

where

K_{max} = *maximum kinetic energy of an electron* (J)

e = *elementary charge* (1.602×10^{-19} C)

V_{stop} = *stopping potential* (V)

Additionally, tests indicate changing the intensity of the incident light has no affect on the kinetic energy of the electrons emitted. This is in contrast to the results expected from the continuous wave equations developed by Maxwell. They suggest increasing wave amplitudes should increase the energy of the electrons emitted, but that's not what's observed. Furthermore, these tests also show varying the frequency of the incident light below a certain point referred to as the cutoff frequency results in no emitted electrons. This result is also unexplained by Maxwell's wave equations where if you supplied enough energy in the form of light intensity, you would expect electrons to be emitted from *T*'s surface [1].

In 1905, Albert Einstein proposed that light exists in discrete amounts and used his theory to explain the results observed from photoelectric effect tests. These discrete amounts would come to be referred to as photons. In Einstein's proposal, a single photon

of light has energy related to its frequency or wavelength conforming to either equation 2 or 3 below:

$$E = hf \tag{2}$$

$$E = (h \cdot c) / \lambda \tag{3}$$

where

E = energy of a single photon (J)

h = Planks constant (6.626×10^{-34} J·sec)

f = frequency of the electromagnetism (cycles/sec)

c = speed of light (2.998×10^{8} m/sec)

λ = *wavelength* (m)

The concept of photons is still poorly understood [1], but to create a model of light as a group of particles occupying space, it would be helpful if their dimensions were defined. One proposed theory on photons provides a range of estimated diameters dependent on the wavelength of the light [2] using Equation 4 below:

$$\lambda/\pi \leq photon\ diameter \leq 10^{4} \cdot \lambda/\pi \tag{4}$$

where

λ = *wavelength* (m)

π = *ratio of a circle's circumference to its diameter* (unitless)

This implies the diameter of photons at common telecommunication wavelengths of 1,310 and 1,550 nanometers might fall in the range of from 4.17×10^{-7} to 4.17×10^{-3} and 4.93×10^{-7} to 4.93×10^{-3} meters, respectively.

2.4 Using Photons For Quantum Key Distribution

Stephen Wiesner originally proposed the idea of using quantum states to store information in his paper Conjugate Pairs [8]. Using the idea of conjugate pairs, Charles Bennett and Gilles Brassard proposed the first QKD protocol in their 1984 paper which has come to be referred to as the BB84 protocol [12]. This protocol uses the polarization directions of single photons to designate binary ones and zeros transferred between participating parties.

To implement the protocol, single photons are created and shared between the communicating parties on what is referred to as the quantum channel. Using the four, non-orthogonal polarization directions of 0, 45, 90, and 135, one of the parties randomly selects a "basis" choice (rectilinear or diagonal) and a bit choice (0 or 1) and polarizes the photon accordingly. The other party randomly selects one of the two bases and measures the photon's polarization. Properties of the conjugate pairs ensure measuring the wrong basis returns a random value and all information stored is subsequently lost. This implies the measuring party will, on average, obtain the correct polarization measurement on half the bits set while the other half, being randomly correct half the time, should be discarded as unreliable. The process of identifying which bits to keep is referred to as sifting.

Sifting requires communication between the two parties over a public channel to exchange the results of their quantum transmissions. Either party can initiate sifting by providing their information to the other. The party setting the polarizations (the setter) can provide the measurer the basis set for each photon. This allows the measurer to exclude the bits either not received at all or measured in the incorrect basis and forward the sifted list to the setter. Conversely, the measurer can send the setter the list of basis

measured and allow the setter to eliminate those bits not received or measured in the incorrect basis and forward the sifted list to the measurer. The reader should note exposing the basis over the public channel in no way reveals the associated bit value.

While a true working implementation of QKD involves much more, the above represents the logical minimum required to perform QKD given ideal components and conditions. Many physical implementations of QKD transmit their photons over optical fiber.

2.5 Optical Fiber

Actual QKD systems frequently use optical fiber as the means of transferring photons between participants. It blocks out external light on the quantum channel, a source of noise for QKD systems operating over open space. In addition, the attenuation of quantum signals is higher in open atmosphere as compared to optical fiber (typically less than 0.35 dB per kilometer [3] and as low as 0.15 dB per kilometer [4]), so quantum signals can travel greater distances through fiber. Finally, quantum signals follow the path of the optical fiber versus the straight path of open space allowing the transfer of quantum signals other than line-of-sight.

Typical optical fiber is made up of three components; the core, cladding, and an exterior coating as illustrated in Figure 4 below:

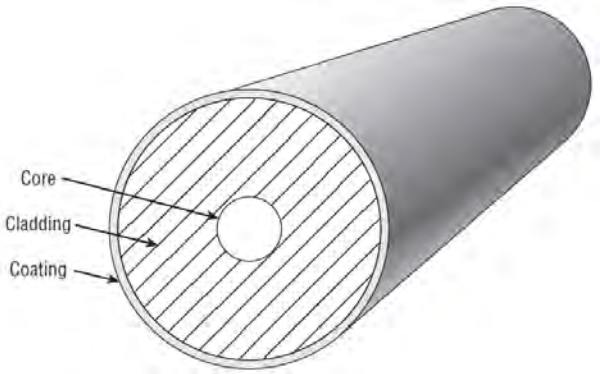

Figure 4: Optical Fiber Construction [3]

The core is meant to carry the bulk of the optical signal and is usually made of very pure glass doped slightly to raise its refractive index. The cladding is also made of pure glass, but manufactured such that its refractive index is lower than the core's. There are numerous different coatings used on optical fiber with all serving the same purpose; to protect the fiber's cladding [3].

Optical fibers work through the process of total internal reflection. Total internal reflection occurs when light passing through the core encounters the lower refractive index of the cladding. This makes the cladding surface reflective and the light bounces off the cladding's internal surface back towards the core.

Typically cylindrical, fiber cores come in various diameters chosen for the signals they're expected to carry as well as the mode they're intended to operate in. Mode refers to the number of paths light can take within an optical fiber with single-mode indicating one path. The mode of operation is a function of the radius of the fiber core, the

wavelength of the light it's carrying, and a difference between the core and cladding's refractive indices know as the numerical aperture [3]. The numerical aperture is given by Equation 5 below:

$$NA = \sqrt{n_{core}^2 - n_{cladding}^2} \tag{5}$$

where

NA = numerical aperture (unitless)

n_{core} = *refractive index of core* (unitless)

$n_{cladding}$ = *refractive index of cladding* (unitless)

The numerical aperture is a measure of how easily an optical fiber accepts light and is always a decimal value between zero and one with zero indicating the fiber accepts no light and one indicating it accepts all light incident on it [3].

Numerical aperture is used in the normalized frequency given by Equation 6 below:

$$V = \frac{2\pi}{\lambda}\, a(NA) \tag{6}$$

where

V = normalized frequency (unitless)

π = *ratio of a circle's circumference to its diameter* (unitless)

λ = *wavelength of light used* (m)

a = radius of the core (m)

NA = numerical aperture (unitless)

To operate in single-mode, a fiber's normalized frequency must fall between 0 and 2.405 [4]. Standard single-mode optical fiber such as ITU-T G.652 has a nominal diameter of 8 to 10 microns and a cladding diameter of 125 microns for use with the common telecommunication wavelengths of 1,310 and 1,550 nanometers [3].

The distribution of light within an optical fiber in single-mode operation is not uniform. The majority of photons traveling in a single-mode optical fiber reside close to the center of the core and diminish in a Gaussian manner [13] out to the cladding whereupon they eventually diminish exponentially [5]. The mode field diameter represents a measure of the light intensity in the cross-section of an optical fiber or a measure of the "transverse electromagnetic field intensity in a fiber cross-section [6]" from plus and minus 1 / e field amplitude points and the 1 / e^2 power points [4]. Figure 5 below illustrates mode field diameter and the corresponding intensity power distribution within optical fiber:

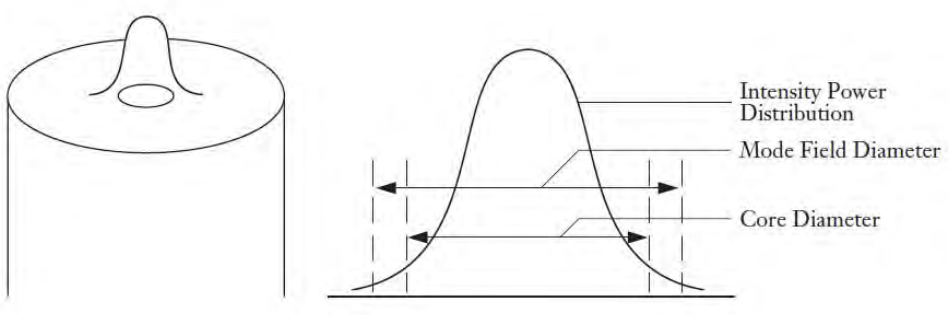

Figure 5: Mode Field Diameter of Single-Mode Optical Fiber [14]

The left of Figure 5 overlays the mode field diameter on top of a cross-section of optical fiber while the right of Figure 5 shows how some of the light passes through the cladding when operating in single-mode. Figure 6 below illustrates this distribution for 1,550 nanometer light in single-mode operation:

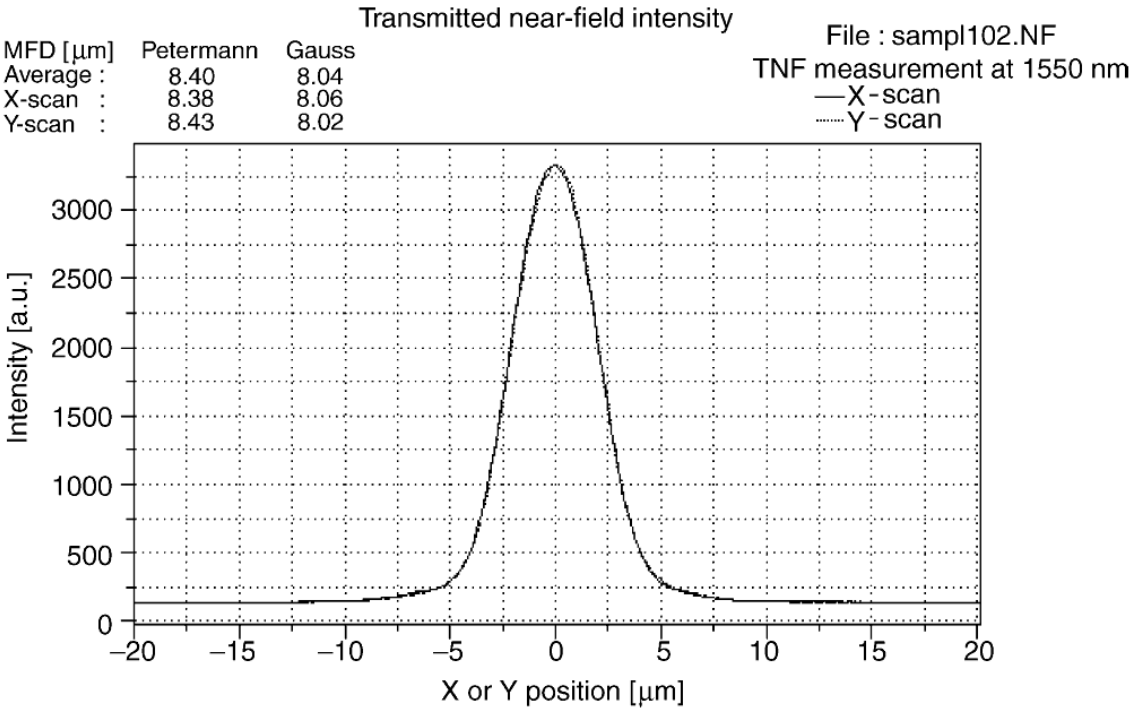

Figure 6: Intensity as a Function of Distance From Core's Center [6]

Figure 6 above shows the greatest intensity occurs at the core's center and decreases in a Gaussian manner out to the cladding where the decrease eventually becomes exponential.

Figure 7 below plots the numerical, Gaussian, and exponential intensities as a function of radial distance in a 9 micron core fiber for 1,550 nanometer light:

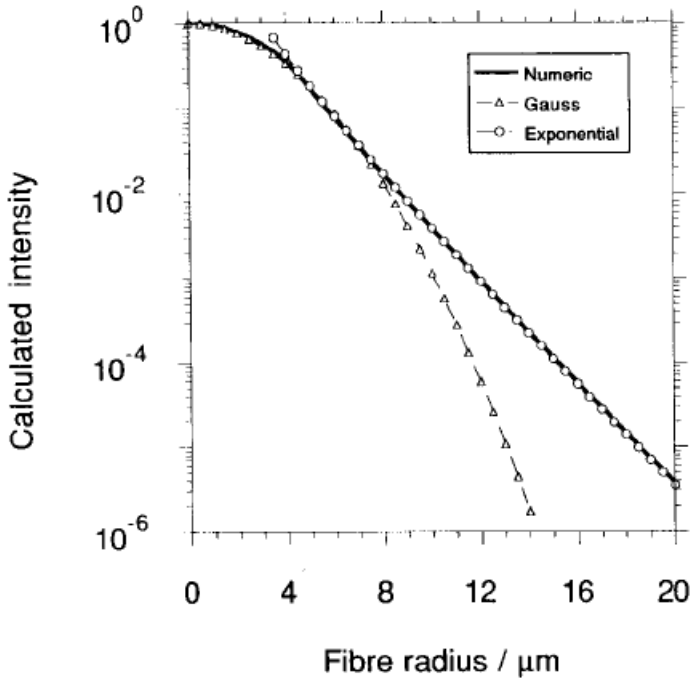

Figure 7: Intensity Comparison as a Function of Distance From Core's Center [5]

Figure 7 above shows the numeric and Gaussian approximations are closest starting from the core's center out to about 7 microns. The exponential approximation starts off divergent until around 4 microns. Into the cladding, the Gaussian starts to diverge as compared to the exponential around 8 microns.

In the core, the Gaussian approximation follows Equations 7 and 8 below [5]:

$$I(r) \sim \frac{1}{\omega_0^2}\, e^{\left(\frac{-2r^2}{\omega_0^2}\right)} \tag{7}$$

$$\frac{\omega_0}{a} = 0.65 + 1.62V^{-3/2} + 2.88V^{-6} \tag{8}$$

where

I = *intensity* (a.u.)

r = *radial distance from core's center* (m)

ω_0 = *mode field radius* (m)

e = *Euler's constant* (unitless)

a = *radius of the core* (m)

V = *normalized frequency* (unitless)

π = *ratio of a circle's circumference to its diameter* (unitless)

λ = *wavelength* (m)

NA = *numerical aperture* (unitless)

Farther into the cladding, the Gaussian approximation is replaced in favor of an exponential approximation using Equations 9 and 10 below [5]:

$$I(r) \sim \frac{1}{r}\, e^{\left(\frac{2vr}{a}\right)} \tag{9}$$

$$v = 1.1428V - 0.996 \tag{10}$$

where

I = *intensity* (a.u.)

r = *radial distance from core's center* (m)

e = *Euler's constant* (unitless)

v = *function of the normalized frequency* (unitless)

a = *radius of the core* (m)

V = *normalized frequency* (unitless)

2.6 Optical Attenuators

In order for QKD to remain perfectly secure as described in the theoretical model, the raw key bit transmitted between parties must be one, and only one, photon. If more than one photon is emitted, a third party could measure one and allow the others to pass by undisturbed in a process known as photon splitting [15]. This process would fundamentally compromise the very purpose of QKD, to permit users the opportunity to detect the presence of an intercepting third party by the perturbations they induce by measuring the transmitted photons. To minimize this, actual QKD implementations must either generate a single photon, or block all but one photon from transmitting [16]. An optical attenuator can accomplish the later.

Normally used to reduce the power level of an incoming signal, an optical attenuator's purpose in a QKD implementation is to cut the signal down so that, optimally, a single photon traverses from transmitter to receiver. The optical attenuator creates losses across itself to reduce the number of photons transmitted at its output. There are typically three ways to realize these losses and provide attenuation; create an air gap or misalignment, absorb photons, or reflect photons [3].

The main losses at an optical junction are caused by misalignments [6] and include:

- x = Lateral offset;
- z = Longitudinal offset;
- ɵ = Angular misalignment; and
- w_T/w_R = Mode field diameter ratio.

Figure 8 below graphically identifies these parameters:

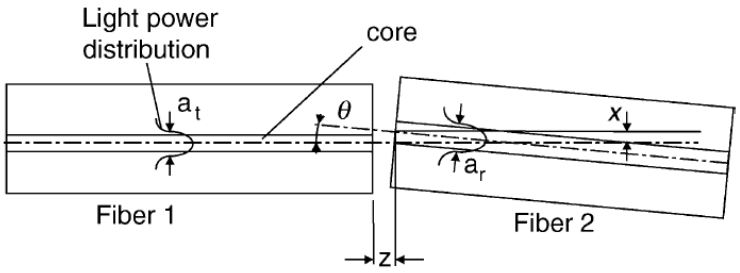

Figure 8: Main parameters affecting joint power loss [6]

From Figure 8 above, we see the lateral offset *x* is the distance between the cores' centers normal to the cores' axes. The longitudinal offset *z* is the distance between the cores' centers parallel to their axes. This is sometimes referred to as an air gap [3]. The angular misalignment θ measures the degree of rotation between the cores' centers relative to their axes. The mode field diameter ratio w_T/w_R is a ratio of the transmitting core's mode field diameter to the receiving core's mode field diameter.

In a single-mode joint with no longitudinal offset, the attenuation is described by Equation 11 below [6]:

$$A(\mathrm{d}B) = -10\log\left\{\left(\frac{2w_{\mathrm{T}}w_{\mathrm{R}}}{w_{\mathrm{T}}^2 + w_{\mathrm{R}}^2}\right)^2 \times \exp\left[-2\frac{(x\lambda)^2 + (\pi n w_{\mathrm{T}} w_{\mathrm{R}}\sin(\theta))^2}{\lambda^2(w_{\mathrm{T}}^2 + w_{\mathrm{R}}^2)}\right]\right\} \tag{11}$$

where A is *attenuation*, w_T is the *transmitting beam mode field radius*, w_R is the *receiving beam mode field radius*, x is the *lateral offset*, λ is the *wavelength*, π is the *ratio of a circle's circumference to its diameter*, n is the *fiber core refractive index*, and θ is the *angular misalignment*. Plots of the attenuation as a function of these parameters is given in Figure 9 below:

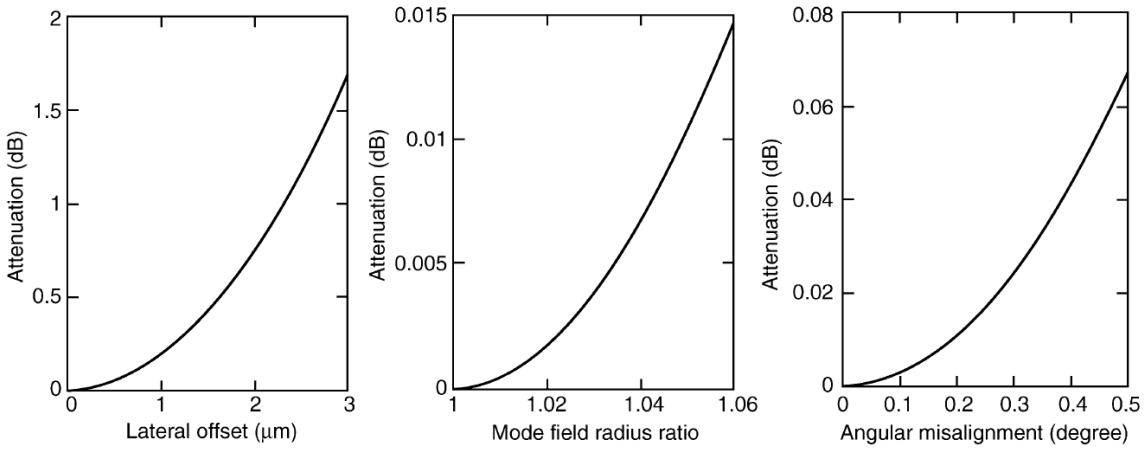

Figure 9: Single-Mode Attenuation as a Function of Geometrical Parameters [6]

Figure 9 shows the main contributor to attenuation in an optical junction is due to lateral offset [6].

Thus, an optical attenuator utilizing geometrical parameters can be implemented by increasing the air gap associated with the longitudinal offset *z*, increasing the lateral offset *x*, or increasing the angular misalignment θ.

Due to differences in the index of refraction between air and an optical core, an optical signal will spread out at an air gap allowing photons to leak out and not pass from the transmitting core to the receiving core. Similarly, photons leak out when a lateral offset exists or the transmitting core is larger than the receiving core.

Before moving on to model the optical attenuator in software, let's review the specific benefits a QKD model, of which the optical attenuator is a necessary component, might provide.

2.7 Computer Simulation

The ultimate purpose of a QKD model is to provide a software abstraction of a physical implementation suitable for simulation. There are numerous advantages to simulating a QKD implementation.

The equipment to implement a QKD network is costly and can require technical expertise to operate, especially when the system implementation is not a commercial product but a laboratory creation.

Individual photons are not directly observable and can travel at the speed of light. A simulation allows analysis at any time for real-time systems or at significant points in

the case of discrete event simulations. Time or events can be sped up or slowed down to collect results that might otherwise take a significant amount of either.

With proper design, system parameters can be varied quickly to alter system configurations and performance.

Finally, a QKD simulation does not have any of the inherently dangerous aspects related to some implementations such as lasers, cryogenic coolers, or associated electrical systems.

In contrast to the above, there are some disadvantages a model simulating a QKD implementation versus the physical implementation itself possesses. The simulation is an approximation of the real implementation and its fidelity is limited. Developing and maintaining a simulation requires not only expertise in implementing the QKD network, but software engineering skills as well.

2.8 Summary

Prior to quantum physics, light was described by Maxwell's equations for electromagnetism and treated as waves. Unexplained observations like the photoelectric effect led Albert Einstein to propose light what made up of discrete particles. These particles would come to be known as photons. Theoretical quantum key distribution can use the polarization of photons to transmit information in a means allowing for the detection of an intercepting party. Current quantum key distribution implementations frequently use lasers and optical attenuators to optimally generate no more than a single photon for each bit of information transmitted over optical fiber in single-mode. The distribution of photons traveling in single-mode optical fiber follows a Gaussian curve

measured by the mode field diameter. Some optical attenuators use geometric misalignments to perform their function. A software model of the interaction between a photon pulse and an optical attenuator would prove useful in simulating certain quantum key distribution implementations. Models of physical components can vary in their representations and how they interact. Tradeoffs between various levels of accuracy and performance must be considered when designing a software model.

III. Methodology

3.1 Overview

This chapter details the simulation environment used and provides descriptions of the model implementations developed for this research. It includes an experiment used to define a dimensional parameter used by the models as well as the simulation experiment used to characterize the models' accuracy.

3.2 Simulation Environment

All simulations were run in 64-bit Windows 7 Professional on a Dell Precision M6500 mobile workstation with an Intel Core i7-920 CPU running at 2.00 GHz and 8 GB of RAM. Simulations are console applications with status information printed in the window they run in. All model coding is in the C programming language and is contained in Appendix A. For random number generation, version 2.1 of the double precision Fast Mersenne Twister developed by Mutsuo Saito and Makoto Matsumoto was used [17].

3.3 Photon Pulse Implementation

A spatial model is considered for the QKD quantum signal where photons are represented as unit particles in a three dimensional matrix where two of the dimensions represent the horizontal and vertical position within the cross-section of an optical fiber and the third dimension represents time in picosecond intervals.

The photon pulse is assumed to arrive at the attenuator within single-mode optical fiber. The cross-section of optical cores and cladding are circular, however, a square model can approximate the same amount of cross-sectional area and make accessing matrix elements easier. This equivalency between a circular versus square model to represent the cross-section of optical fibers is illustrated graphically in Figure 10 below:

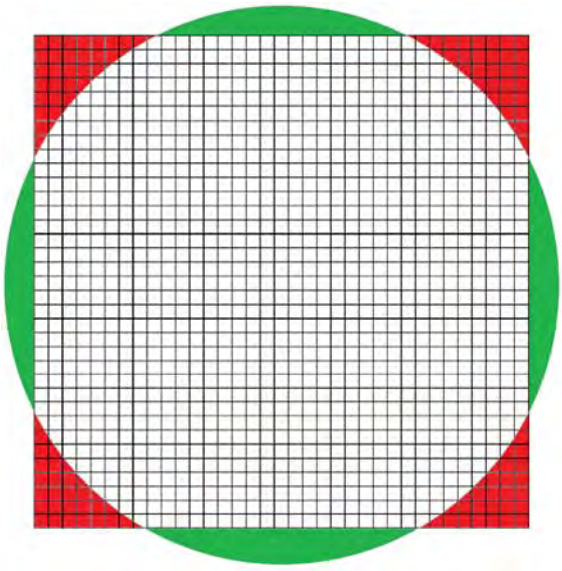

Figure 10: Demonstrating Equivalent Areas Between Square and Circular Models

with equivalency occurring when the areas in red and green equal one another.

The square model's dimensions for equivalency are determined by setting the equations for the area of a square equal to the area of a circle as in Equation 12 below:

$$x^2 = \pi \cdot r^2 \tag{12}$$

where

x = length of the square's side (m)

r = length of the circle's radius (m)

For example, using a radius of 4×10^{-6} meters for the assumed optical fiber core for single-mode operation, the equivalent square's sides would be determined using Equation 13 below:

$$x = \sqrt{\pi(4 \times 10^{-6})^2} \tag{13}$$

Solving for *x*, an equivalent square model's sides become 7.09×10^{-6} meters.

Given this square matrix representation, an arbitrary frame of reference is selected and labeled such that in the matrix, the vertical dimension increases from 0 to some defined upper bound from top to bottom to indicate rows and the horizontal dimension increases from 0 to the upper bound from left to right to indicate columns as illustrated in Figure 11 below:

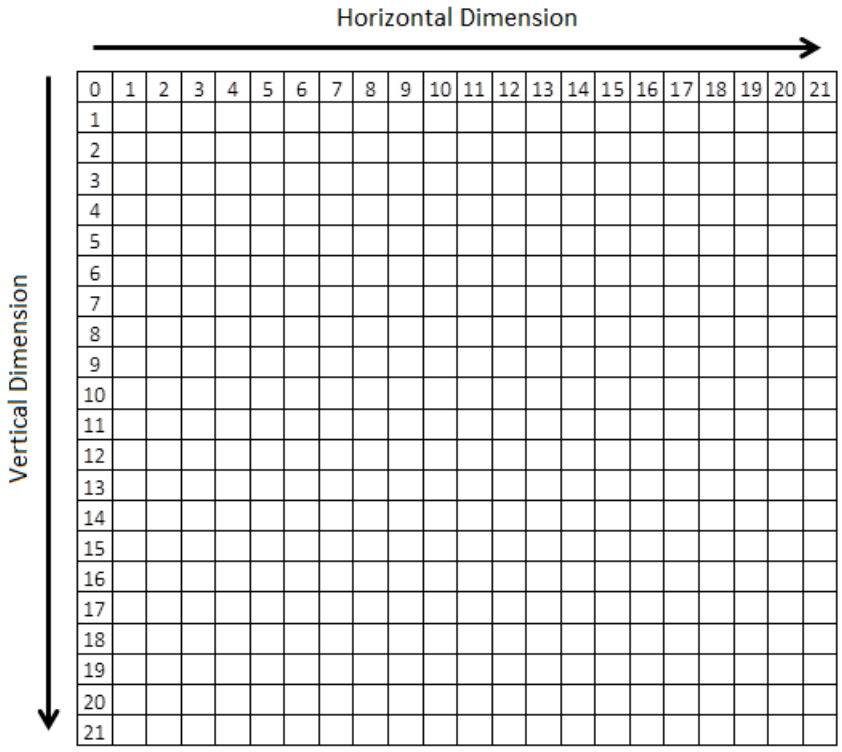

Figure 11: Frame of Reference for Labeling Horizontal and Vertical Matrix Dimensions

Figure 11 above illustrates an example matrix with the maximum dimension defined as 22 such that a 22 by 22 matrix is defined.

The next step to consider is how to identify the upper bound dimension so as to divide the matrix into discrete elements to contain the quantum signal's photons. Several important characteristics must be considered to determine the number of elements to use; how should the total photon count be distributed amongst the elements, what is the highest number of photons expected in a single picosecond from the quantum signal

source, and can an offset attenuator using the photon pulse return photon counts representative of current QKD implementations?

First, the distribution is considered. The overall intensity of the photon pulse for any given picosecond is distributed within the cross-section of an optical fiber in a Gaussian manner fitting Equation 7 above. One means of accomplishing this distribution is to divide the matrix's horizontal and vertical elements into concentric regions and fill each region with a representative percentage of the total number of photons for that picosecond consistent with the mode field diameter distribution as illustrated in Figure 12 below:

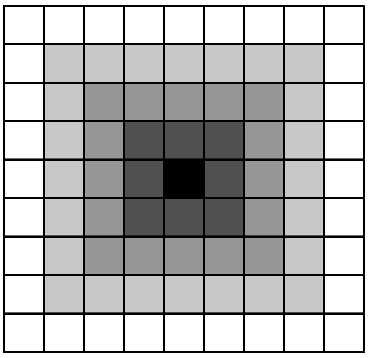

Figure 12: Increasing Photon Density Towards Fiber Core's Center

Figure 12 above shows a 9 by 9 element matrix defining 5 concentric regions where the outer ring of elements represents the outermost region with the least intensity

and consequently the least shading. Each concentric region has higher intensity and correspondingly darker shading as we move towards the innermost region containing the highest concentration of photons and thus the darkest shade.

However many regions are used, the outermost region is defined to contain the remaining photon percentage not allocated to the inner regions by the Gaussian distribution. This allows the model to account for the increasingly smaller percentage of intensity as the Gaussian's limit tends towards zero.

The next characteristic to consider is representing the maximum photon count for a single picosecond. Figure 13 below illustrates the intensity of a generic laser pulse with respect to time:

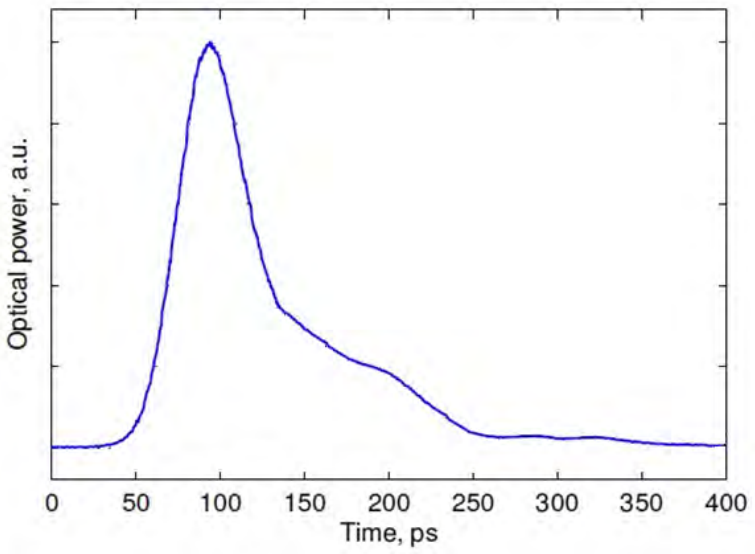

Figure 13: Generic Laser Pulse Shape As a Function of Time

Using an oscilloscope of sufficient resolution, it is possible to sample a laser pulse like the one in Figure 13 above at picosecond intervals and convert the measured power into photon count by dividing by the energy of a single photon given in Equation 3 above and create a photon pulse profile of picosecond resolution. Storing such a photon pulse profile as the total number of photons to emit per picosecond in a file permits profiling multiple different quantum sources and provides the starting basis for an algorithmic means of implementing the photon pulse model.

The photon pulse is added to the three dimensional matrix one picosecond at a time starting with reading the file and obtaining the total photon count for the current picosecond. That picosecond's total number of photons is then distributed into the matrix starting with the outermost region and working towards the innermost region.

Each region is assigned a percentage of the total photon count for the current picosecond based off the number of regions modeling the photon pulse and distributed in conformance with the mode field diameter. To do this, a cumulative distribution function is used to integrate the intensity function defined by Equation 7 above for each region. The resulting percentages define the portion of the entire photon pulse to assign to each region by multiplying each region's percentage with the total photon count to generate the total number of photons for each region, respectively. If the number of photons for a region falls below one, a single check of probability is used to determine if one photon is placed in that region.

The capacity of a region is defined as the number of elements in the region times the integer represented by the unsigned element size used, such as 255 for a single byte of storage per element or 65,535 for two bytes of storage. Before placing photons in the

region, the number of photons to place is compared to the capacity of the region to ensure there's storage for this region's photons. If not, a notification is generated informing the user the total photon count for the current picosecond called for from the photon pulse profile input file exceeds the model's capabilities and the region is filled to capacity.

Otherwise, starting with the upper left element of the region, each element in the region is checked probabilistically to see if it contains the next photon to place in a counter-clockwise manner until all photons for the region are placed. The probability of placing a photon in the element is a function of the total number of photons to place in the region divided by the number of elements in the region. This process of probabilistically determining whether an element receives the next photon to place is to avoid merely placing photons in order which would skew results for attenuation events containing the left and bottom sides of a region which would get photons before the right and top sides of the region. During this rotation, individual elements are checked to confirm they still possess storage capacity for another photon. If not, the element is skipped and the next element is considered. This process repeats until the region's photons are fully placed at which point the process repeats with the next region until all the photons for this picosecond are placed. Figure 14 below illustrates the process for the outermost and next regions in an example 22 by 22 matrix:

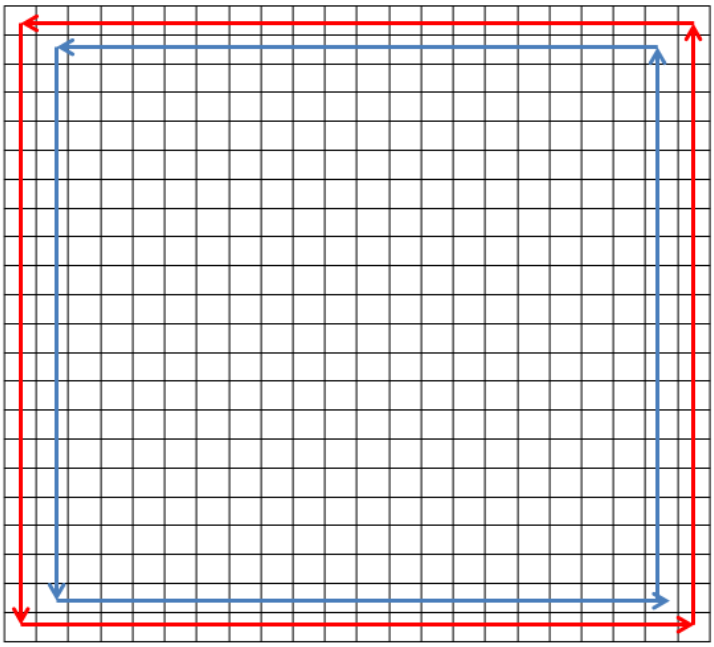

Figure 14: Path For Filling Regions

The red arrows in Figure 14 above illustrate the circuit followed in the outermost region starting in element (0, 0) in the upper left hand corner per the frame of reference selected previously and proceeds downward to (1, 0), (2, 0), etc. until reaching element (21, 0) when it starts moving rightward to (21, 21), then upward to (0, 21), then leftward to (0, 0) in a circuit filling elements with photons. The circuit continues until all photons designated for the region are placed whereupon the next concentric region, identified by the blue arrows, is filled starting with its upper left element at (1, 1) and so forth moving towards the innermost regions until the entire photon pulse for the current picosecond is placed.

A slightly different process is used for the innermost region. The matrix's dimensions are defined to be odd such that the innermost region always contains 9 elements instead of 4 as demonstrated by Figure 15 below:

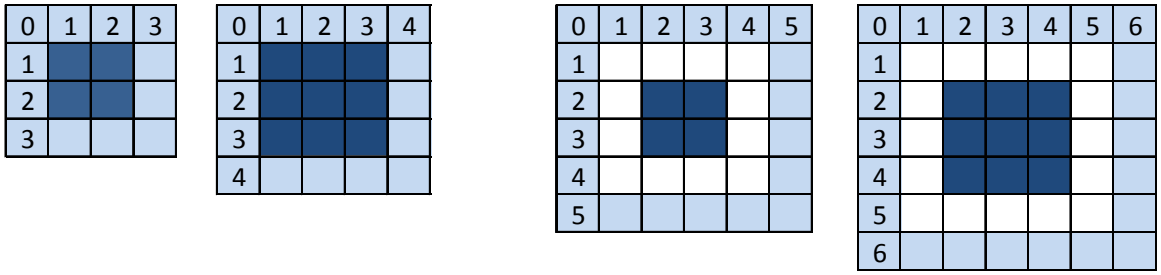

Figure 15: Example Matrices Demonstrating Equal Numbers of Regions

Figure 15 above shows two pairs of matrices with the same number of regions, but varying numbers of innermost elements defined. The pair on the left both have two regions defined; the innermost region highlighted in dark blue and the outermost region highlighted in light blue. The matrix with odd dimensions of 5 (0 through 4) elements is used as the innermost region contains 9 elements instead of 4. Similarly, the pair of matrices on the right of Figure 15 have three regions defined; the innermost region highlighted in dark blue, one unhighlighted intermediate region, and the outermost region highlighted in light blue. The matrix with odd dimensions of 7 elements is used as the innermost region contains 9 elements instead of 4.

Instead of filling the innermost region elements in a circuitous manner as before, elements are added to the 3 rows top to bottom, left to right repetitively until the remaining photons are placed.

After the total number of photons for the picosecond are placed in their respective regions in a manner consistent with the mode field diameter distribution, the next picosecond of the photon pulse is read from the photon pulse profile input file and the previous processes repeat. After the last picosecond is placed, the entire photon pulse is complete and, for this research, submitted to the optical attenuator for attenuation.

By distributing the photon pulse in the manner described above, the author proposes a reasonably accurate spatial model of the laser pulse as discrete particles is created suitable for processing by a lateral offset attenuator using geometric misalignments to determine which photons are attenuated and which photons pass on.

3.4 Lateral Offset Attenuator Implementation

The photon pulse representation described above as a spatial grouping of photons in three dimensions permits an attenuator that uses the concept of lateral misalignment between two abutted faces of an optical junction for creating the net attenuation.

Following the model described for the photon pulse above, the opposing faces of the abutted junctions are segmented into the same number of discrete elements representing channels within the cross-section of the optical fiber used to define the location of photons.

A horizontal and vertical parameter are defined to indicate the amount of lateral offset to apply. Using the spatial information contained in the three dimensional matrix

representation of the photon pulse, the attenuator passes photons that match up from face to face and drops those which are misaligned. Increasing lateral offset is defined as dropping channels from bottom to top and from right to left. Figure 16 below illustrates an example 22 by 22 matrix with offset settings of vertical = 18 and horizontal = 18 which would result in the attenuation of all photons propagating in the shaded channels:

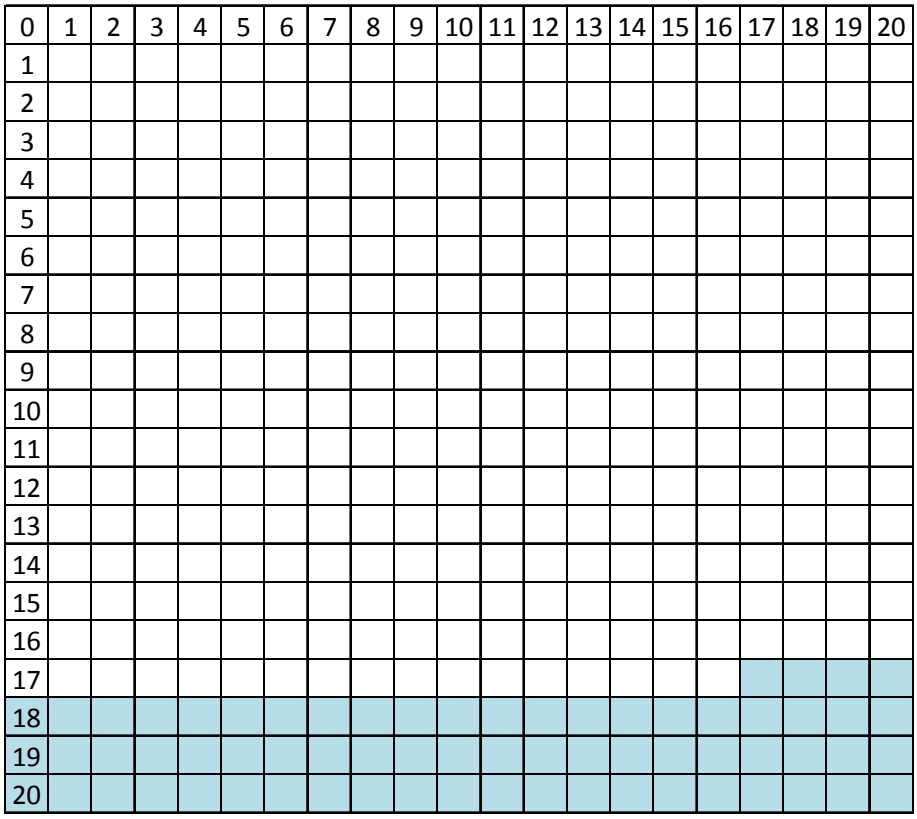

Figure 16: Lateral Offset Example Demonstrating Attenuated Elements

Algorithmically, the lateral offset attenuator works by setting the elements of the pulse matrix that lie outside the offset bounds to zero for each picosecond of the entire photon pulse duration. Setting the horizontal offset parameter to the matrix's maximum dimension or more results in no attenuation of the starting row indexed by the vertical offset parameter. Setting the vertical offset parameter to the matrix's maximum dimension or more implements no attenuation of the input photon pulse.

3.5 Experiment 1: Define Photon Pulse Dimensions

3.5.1 Objective

The objective of Experiment 1 is to identify the number of regions, and consequently the horizontal and vertical dimensions, necessary to accommodate a representative QKD quantum signal source in conformance with the characteristics of the chosen quantum signal source, the optical fiber used, the mode field diameter distribution, and the lateral offset attenuator's expected functionality.

3.5.2 Parameters

Parameters associated with Experiment 1 include the characteristics of the components involved including the quantum signal source and the optical fiber used as detailed below:

1. **Number of regions** - the number of equally sized concentric regions to divide a single picosecond of quantum signal into. Does not include an extra outermost region to catch the remaining photons as the limit of the Gaussian approximation tends towards zero.

2. **Outer Region Elements** - the number of photon pulse matrix elements in the outermost region and a function of the number of regions.

3. **Inner Region Elements** - the number of photon pulse matrix elements in the innermost region. Always defined as 9 by using an odd number for the matrix's dimensions.

4. **Wavelength** - the wavelength of the quantum signal source used.

5. **Peak Source Power** - the maximum power supplied by the quantum signal source.

6. **Source Duration** - the time difference from start to finish of the quantum signal source.

7. **Radial Distance** - a measure of the distance from the fiber's center normal to its axis.

8. **Outer Radial Limit** - the radial distance modeled by the number of regions.

9. **Interval** - the size of each region given by the outer radial limit divided by the number of regions.

10. **Core's Radius** - the radial distance from the fiber's center where the fiber's core ends and the fiber's cladding begins.

11. **Core's Index of Refraction** - ratio of the speed of light in a vacuum to the speed of light in the fiber's core.

12. **Cladding's Index of Refraction** - ratio of the speed of light in a vacuum to the speed of light in the fiber's cladding.

13. **Numerical Aperture** - measure of how easily the fiber accepts light as defined by Equation 5 above.

14. **Normalized Frequency** - the sum of the squares of the eigenvalues in the core and cladding defined by Equation 6 above.

15. **Mode Field Radius** - half the mode field diameter and a function of the normalized frequency and the fiber core's radius.

16. **Intensity** - the quantum signal's field strength squared and approximated by a Gaussian function defined by Equation 7 above.

17. **Standard Deviation** - measure of variability defined as half the mode field radius.

18. **Element Size** - the number of bytes used to represent the number of photons an individual matrix element contains.

3.5.3 Methodology

Using the equations defining the applicable parameters above, a Matlab script is used to iterate through various combinations of parameters to calculate intensity distributions and corresponding region percentages of the total photon pulse count for a given picosecond to identify photon pulse configurations suitable for modeling the quantum signal's photon pulse. The script is presented in Appendix B.

The parameters associated with the quantum source and optical fiber are kept fixed and are summarized in Table 1 below:

Table 1: Experiment 1 Fixed Parameters

Parameter	Setting
Inner Region Elements	9
Wavelength	1,310 nm
Peak Source Power	0.001 J/sec
Source Duration	400×10^{-12} sec
Core radius	4×10^{-6} m
Core's Index of Refraction	1.49
Cladding's Index of Refraction	1.485
Numerical Aperture	0.12196
Normalized Frequency	2.3399
Mode Field Radius	4.4806×10^{-6} m
Maximum Photon Count Per Picosecond	6,595 photons
Element Size	1 byte

While the script does calculate some of the fixed parameters such as numerical aperture and normalized frequency, their values don't change between iterations for Experiment 1 and are strictly for maintaining the generality of the script for implementing other quantum signal source and fiber permutations.

The parameters to be varied include the outer radial limit and the number of regions which, by definition, varies the number of elements in the outermost region.

Using these parameters, the script calculates the interval, an array of radial distances from 0 to the outer radial limit in interval amounts, and the intensity at each of the defined radial distances. Then, using a cumulative distribution function, a percentage for each defined radial distance is calculated along with the remaining percentage covered by the outermost region to account for the entire Gaussian approximation.

The outer radial limit starts at the fiber core's radius of 4 microns, given the bulk of the quantum signal is carried in the fiber's core, and the data for several incremental micron steps are collected and compared to establish the impact of modeling more and more of the fiber's cladding.

The number of regions is incremented from 1 in 1 region increments until the number of photons for the innermost region reaches a number for which the 9 innermost elements are capable of storing their region total based on their percentage of the total photon pulse and the element size used to store photons.

For each number of regions increment, the probability of returning a single photon from a single element in the outer region is calculated to compare against the desired result of 10 percent.

3.5.4 Assumptions and Limitations

The quantum signal's source generating photon pulses is fixed as a 400 picosecond duration laser pulse with a maximum peak power output of 1 milliwatt and a wavelength of 1,310 nanometers. Assuming such a source, the maximum photon count for a single picosecond is determined using Equation 3 above. Starting with the energy of a single photon as $(6.626 \times 10^{-34}$ J·sec · 2.998×10^{8} m/sec) / 1310×10^{-9} m yields 1.516×10^{-19} J/photon. Dividing the assumed quantum signal source peak output power

of 1 milliwatt by the energy of a single photon results in (0.001 J/sec · 10^{-12} sec/picosecond) / 1.516 J/photon for a total of 6,594.5966 photons/picosecond rounded up to 6,595 photons/picosecond.

The photon pulse is assumed to originate within a step indexed optical fiber operating in single-mode. The parameters identified in Table 1 above are selected to accomplish such.

The overall size of the photon pulse matrix is limited by the amount of memory supplied by the simulation environment. Given an unsigned element of two bytes storage can accommodate 65,535 photons on its own, only unsigned elements of single byte storage capable of storing 255 photons each are considered.

3.5.5 Expected Results

Some combinations of parameters will result in photon pulse model dimensions not suited for representing the quantum source parameters used either due to not being able to contain the peak number of photons in the innermost region or for having too many possible photons in the outermost region. Successful photon representations support the expected functionality of a QKD lateral offset attenuator by returning a single photon from a single outermost region element for the quantum signal's duration 10 percent of the time and represent the distribution of photons in a manner consistent with the mode field diameter by containing an appropriate percentage of the total photon pulse in each defined region including the peak photon counts expected from the quantum signal source.

The way in which the lateral offset attenuator's functionality is defined is such that the maximum amount of attenuation short of complete blocking of all photons allows

a single element (0, 0) per picosecond to pass unattenuated. A successful photon pulse representation must allow no more than this single element multiplied by the photon pulse duration to contain no more than the expected average photon count of 0.1 photons. As there are no such things as partial photons, a 10 percent probability of a single photon being placed in the (0, 0) matrix element over the pulse duration is used to denote acceptable attenuator functionality. This probability is calculated by dividing the number of photons in the outermost region by the number of elements in the region for a single picosecond and multiplying by the photon pulse's duration.

The distribution of photons within a single picosecond is defined by the mode field diameter with the peak number of photons occurring near the center of the fiber and diminishing in a Gaussian manner with radial distance from the fiber's center. While the intensity shifts from a Gaussian to exponential approximation, this doesn't occur until around 8 microns from the core's center where the percentage of photons starts seeing a steeper drop off in the Gaussian approximation. A successful photon pulse representation is defined as one that models this distribution through the entirety of the photon pulse.

The number of photon pulse matrix elements is limited by the simulation environment. Furthermore, the number of elements per region diminishes moving from the outermost region towards the innermost region while the number of photons per region increases towards the innermost region to support the Gaussian distribution. A successful photon pulse representation will support the peak power output of the quantum signal source by containing the maximum number of photons in the inner region consistent with the mode field diameter percentages obtained from the cumulative distribution of the Gaussian intensity defined for the innermost region.

3.6 Experiment 2: Characterize Attenuator Accuracy

3.6.1 Objective

Experiment 2 looks at characterizing the accuracy of the software model of a lateral offset attenuator using the photon pulse model established from the results of Experiment 1. The output of the model is compared against the mathematically expected results for varying amounts of lateral offset.

3.6.2 Simulation Parameters

Since the lateral offset attenuator uses the photon pulse established in Experiment 1, Experiment 1's parameters are also applicable to Experiment 2. The following additional parameters are also applicable:

1. **Vertical Offset Setting -** an index into the three dimensional matrix defining the row where attenuation starts.

2. **Horizontal Offset Setting -** an index into the three dimensional matrix defining the element within the vertical offset setting's row where attenuation starts.

3. **Number of Test Iterations -** the number of attenuation events to conduct at the given vertical and horizontal offset setting pair.

3.6.3 Methodology

Experiment 2 uses the same quantum signal and optical fiber characteristics established in Experiment 1, however, the total photon count for each picosecond is now defined by a photon pulse profile input file representative of Figure 13 above.

For Experiment 2, all parameters defined in Experiment 1 are fixed for Experiment 2.

The variable parameters for Experiment 2 include the lateral offset attenuator's vertical and horizontal offset settings as well as the number of test iterations.

To test the accuracy of the lateral offset attenuator model, a quantum signal source provides an identical photon pulse to the attenuator defined in a photon pulse profile file with 400 picoseconds of photon counts roughly matching the photon pulse profile given in Figure 13 above. The total photon count for all picoseconds in the file is 581,730 photons.

Next, the attenuator's vertical and horizontal offset settings are incremented from zero to their maximum values in increments of 1. At each setting, the average number of photons passed through the attenuator unattenuated is recorded for 1, 10, 100, and 1,000 test iterations to view statistical variance between increasing numbers of attenuation events. The resulting attenuated photon count is converted to total energy by multiplying by the energy of a single photon defined by Equation 7 above and then to power by dividing by the pulse's duration. The resulting power is converted to decibels and compared to the expected attenuation amounts calculated using the simplified equation for attenuation defined by Equation 11 above.

3.6.4 Assumptions and Limitations

The photon pulse model is assumed to simulate an actual QKD source closely enough to allow accurate attenuation results with the accompanying assumption the lateral offset attenuator model accurately simulates the behavior of a real world lateral offset attenuator implemented as functioning by means of geometrical misalignments.

The photon pulse is assumed to arrive at the attenuator through step indexed optical fiber operating in single-mode and individual photons traveling through its cross-section are assumed to remain within their discrete channel as they cross the optical junction and not migrate between channels until after the junction.

Using an unsigned element size of one byte, the number of photons per element in the matrix is limited to 255. The overall size of the matrix is limited by the simulation environment.

3.6.5 Expected Results

While there are several assumptions and approximations incorporated into both the photon pulse and attenuator models reducing their fidelity from the actual physical processes governing the interactions observed with real physical systems, it's expected the software models represent these physical systems enough to generate results in line with the mathematical calculations derived from Equation 11 above. The random elements associated with generating the photon pulse are expected to produce minimal variance in the final attenuated photon pulse.

A successful experiment will result in a number of attenuation events throughout the range of the attenuator's lateral offset settings by which the accuracy of the attenuator model will be characterized by statistical analysis.

3.7 Summary

This chapter started with a summary of the simulation environment and went on to describe a three dimensional particle based model for representing a QKD quantum signal source as well as an algorithmic process for developing one. Taking this pulse

model as a basis, an optical attenuator using lateral offset misalignments was described. An experiment to derive efficient dimensions for modeling the photon pulse was presented along with an experiment to characterize the accuracy of the attenuator using the photon pulse model.

IV. Analysis and Results

4.1 Overview

This section details the empirical results obtained from Experiments 1 and 2 along with an analysis of their impact on the research questions posed in this paper.

4.2 Experiment 1 Results

Tables 2 through 7 below present the results of Experiment 1 with an increasing outer radial limit and number of regions until the last row indicates an outer radial limit/number of regions pair producing an innermost region capable of containing the photon counts expected at peak power from the quantum signal source:

Table 2: Experiment 1 Results - Outer Radial Limit = 4×10^{-6} m

Outer Radial Limit (m)	Number of Regions	Innermost Region Percentage	Outermost Region Percentage	Number of Outer Elements	Innermost Region Photons	Outermost Region Photons for One Element
$4x10^{-6}$	1	92.581511	7.418489	16	6105.750650	30.578084
$4x10^{-6}$	2	62.800126	7.418489	24	4141.668310	20.385390
$4x10^{-6}$	3	44.826155	7.418489	32	2956.284922	15.289042
$4x10^{-6}$	4	34.466844	7.418489	40	2273.088362	12.231234

Table 3: Experiment 1 Results - Outer Radial Limit = 6×10^{-6} m

Outer Radial Limit (m)	Number of Regions	Innermost Region Percentage	Outermost Region Percentage	Number of Outer Elements	Innermost Region Photons	Outermost Region Photons for One Element
$6x10^{-6}$	1	99.259825	0.740175	16	6546.185459	3.050909
$6x10^{-6}$	2	81.946302	0.740175	24	5404.358617	2.033939
$6x10^{-6}$	3	62.800126	0.740175	32	4141.668310	1.525454
$6x10^{-6}$	4	49.685642	0.740175	40	3276.768090	1.220364
$6x10^{-6}$	5	40.779359	0.740175	48	2689.398726	1.016970
$6x10^{-6}$	6	34.466844	0.740175	56	2273.088362	0.871688

Table 4: Experiment 1 Results - Outer Radial Limit = 8×10^{-6} m

Outer Radial Limit (m)	Number of Regions	Innermost Region Percentage	Outermost Region Percentage	Number of Outer Elements	Innermost Region Photons	Outermost Region Photons for One Element
$8x10^{-6}$	1	99.964430	0.03557	16	6592.654159	0.146615
$8x10^{-6}$	2	92.581511	0.03557	24	6105.750650	0.097743
$8x10^{-6}$	3	76.607709	0.03557	32	5052.278409	0.073308
$8x10^{-6}$	4	62.800126	0.03557	40	4141.668310	0.058646
$8x10^{-6}$	5	52.488940	0.03557	48	3461.645593	0.048872
$8x10^{-6}$	6	44.826155	0.03557	56	2956.284922	0.041890
$8x10^{-6}$	7	39.004288	0.03557	64	2572.332794	0.036654
$8x10^{-6}$	8	34.466844	0.03557	72	2273.088362	0.032581

Table 5: Experiment 1 Results - Outer Radial Limit = 9×10^{-6} m

Outer Radial Limit (m)	Number of Regions	Innermost Region Percentage	Outermost Region Percentage	Number of Outer Elements	Innermost Region Photons	Outermost Region Photons for One Element
$9x10^{-6}$	1	99.994113	0.005887	16	6594.611752	0.024265
$9x10^{-6}$	2	95.542629	0.005887	24	6301.036383	0.016177
$9x10^{-6}$	3	81.946302	0.005887	32	5404.358617	0.012133
$9x10^{-6}$	4	68.477928	0.005887	40	4516.119352	0.009706
$9x10^{-6}$	5	57.829231	0.005887	48	3813.837784	0.008088
$9x10^{-6}$	6	49.685642	0.005887	56	3276.768090	0.006933
$9x10^{-6}$	7	43.396555	0.005887	64	2862.002802	0.006066
$9x10^{-6}$	8	38.444771	0.005887	72	2535.432647	0.005392
$9x10^{-6}$	9	34.466844	0.005887	80	2273.088362	0.004853

Table 6: Experiment 1 Results - Outer Radial Limit = 10×10^{-6} m

Outer Radial Limit (m)	Number of Regions	Innermost Region Percentage	Outermost Region Percentage	Number of Outer Elements	Innermost Region Photons	Outermost Region Photons for One Element
$10x10^{-6}$	1	99.999194	0.000806	16	6594.946844	0.003322
$10x10^{-6}$	2	97.437441	0.000806	24	6425.999234	0.002215
$10x10^{-6}$	3	86.322083	0.000806	32	5692.941374	0.001661
$10x10^{-6}$	4	73.554345	0.000806	40	4850.909053	0.001329
$10x10^{-6}$	5	62.800126	0.000806	48	4141.668310	0.001107
$10x10^{-6}$	6	54.309110	0.000806	56	3581.685805	0.000949
$10x10^{-6}$	7	47.631040	0.000806	64	3141.267088	0.000831
$10x10^{-6}$	8	42.312816	0.000806	72	2790.530215	0.000738
$10x10^{-6}$	9	38.008038	0.000806	80	2506.630106	0.000664
$10x10^{-6}$	10	34.466844	0.000806	88	2273.088362	0.000604

Table 7: Experiment 1 Results - Outer Radial Limit = 11×10^{-6} m

Outer Radial Limit (m)	Number of Regions	Innermost Region Percentage	Outermost Region Percentage	Number of Outer Elements	Innermost Region Photons	Outermost Region Photons for One Element
$11x10^{-6}$	1	99.999909	0.000091	16	6594.993999	0.000375
$11x10^{-6}$	2	98.591245	0.000091	24	6502.092608	0.000250
$11x10^{-6}$	3	89.830308	0.000091	32	5924.308813	0.000188
$11x10^{-6}$	4	78.036962	0.000091	40	5146.537644	0.000150
$11x10^{-6}$	5	67.390471	0.000091	48	4444.401562	0.000125
$11x10^{-6}$	6	58.683752	0.000091	56	3870.193444	0.000107
$11x10^{-6}$	7	51.696852	0.000091	64	3409.407389	0.000094
$11x10^{-6}$	8	46.062328	0.000091	72	3037.810532	0.000083
$11x10^{-6}$	9	41.463205	0.000091	80	2734.498370	0.000075
$11x10^{-6}$	10	37.657682	0.000091	88	2483.524128	0.000068
$11x10^{-6}$	11	34.466844	0.000091	96	2273.088362	0.000022

For each of Tables 2 through 7, the first column represents the outer radial limit variable parameter. Each table summarizes its data for this parameter, respectively. The second column presents the number of regions variable parameter starting from 1 and incrementing by 1 until the number of innermost photons in column 6 reaches a level representable by the innermost region's storage capacity. The third column identifies the percentage of the total photon count for a single picosecond at peak power the innermost region must contain to accurately represent the mode field diameter profile. The fourth column identifies the percentage of the total photon count for a single picosecond at peak power the outermost region must contain to accurately represent the mode field diameter

profile. The fifth column identifies the number of individual elements in the outermost region given the number of regions defined in column 2. The sixth column identifies the number of photons the innermost region must contain to accurately represent the mode field diameter profile. The seventh column identifies the average number of photons contained in a single element of the outermost region for a single picosecond at peak power.

The innermost region of the photon pulse is always the most limiting as compared to any other region's photon capacity as it always contains the highest percentage of photons and the lowest number of elements. By design, the innermost region always contains 9 elements. Multiplying each of these elements by the storage capacity of an unsigned element size of 1 byte results in a peak photon capacity of 9 elements $\times$ 255 photons/element or 2,295 photons the innermost region can accommodate. It was previously shown the quantum signal at peak power generates 6,595 photons/picosecond. Column 6 of each table results from multiplying this peak photon count by column 3's percentage. Of note for each table, the innermost region becomes viable when the number of regions equals the outer radial limit's mantissa with the photon count consistently reaching 2273.088362 photons in all cases as demonstrated in Figure 17 below:

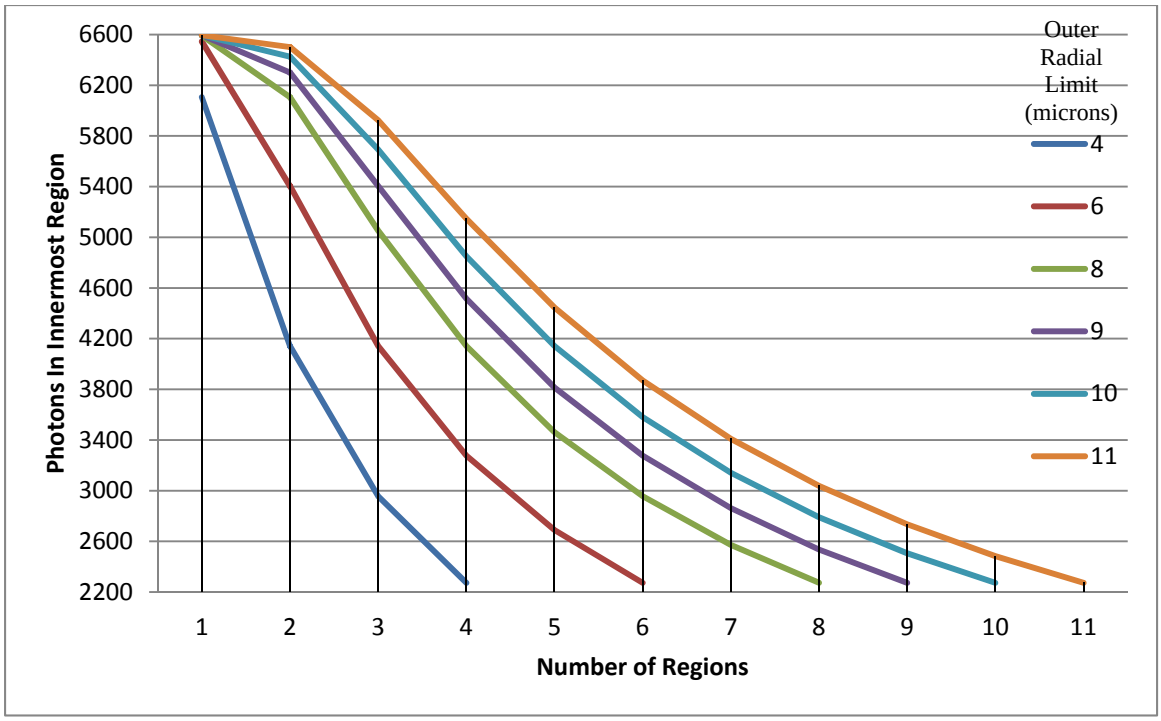

Figure 17: Photon Count In Innermost Region as a Function of Outer Radial Limit and Number of Regions

Figure 17 illustrates how increasing outer radial limit and number of regions results in photon counts more consistently resembling the Gaussian distribution approximation used for calculating intensity.

Column 4 of each table shows the percentage of photons in the outermost region is strictly a function of the outer radial limit used. Of note for each table, while column 7 shows that increasing the number of regions and subsequently the number of outermost region elements decreases the average photon per element in the outermost region, the effect is significantly smaller than increasing the outer radial limit. Even with 500

regions defined, the average photon count per element only falls to 0.122 when the outer radial limit is equal to the fiber core's radius.

Instead, an efficient balance between outer radial limit and number of regions is found by observing the last entry of column 7 for the viable representations shown in the last row of each table. In the case of Tables 2 and 3, the average photon count for a single element in the outermost region stays above the target 0.1 at 12.23 and 0.87, respectively. These are obviously poor candidates. Table 4 with an outer radial limit of 8 microns, or double the fiber core's radius, is the first candidate with a column 7 value below the target value of 0.1 at 0.03. While this is certainly a fully viable pair of outer radial limit and number of regions parameters to support the quantum signal's peak power, the average in column 7 is for a single picosecond. The remaining 399 picoseconds in the quantum signal's pulse duration may add enough photons to the average such that the total exceeds the target 0.1 per quantum signal.

To resolve the entire pulse duration, a worst case scenario of peak power over the entire pulse duration can be assumed. Now, the average photon count for a single picosecond times the pulse duration becomes 0.032581 photons/picosecond × 400 picoseconds or 13.0324 photons. This makes the pair of outer radial limit of 8 microns and 8 number of regions parameters unacceptable for the worst case scenario. Similar calculations for 9 and 10 micron outer radial limits with 9 and 10 number of regions respectively results in average photon counts of 1.9412 and 0.2416, respectively.

It is not until the outer radial limit of 11 microns with 11 number of regions occurs in Table 7 that a value of 0.000022 photons/picosecond × 400 picoseconds or 0.0088 photons is attained and a fully compliant photon pulse representation able to

handle the quantum signal's peak output power with its innermost region and an outermost region able to return less than the target photon count of 0.1 per quantum signal is identified.

The resulting region percentages for this outer radial limit and number of regions parameter pair are detailed in Table 8 below which provides a region by region breakdown of percentages to accurately model the mode field diameter distribution with the defined quantum source and optical fiber assumptions of Experiment 1:

Table 8: Percentage of Total Photon Count By Region For Outer Radial Limit and Number of Regions Equal to 11

Region's Radial Range (m)	Percent of Total Photon Count
0.00 to 1.00 $x10^{-6}$	34.466844
1.00 to 2.00 $x10^{-6}$	28.333282
2.00 to 3.00 $x10^{-6}$	19.146176
3.00 to 4.00 $x10^{-6}$	10.635208
4.00 to 5.00 $x10^{-6}$	4.855931
5.00 to 6.00 $x10^{-6}$	1.822385
6.00 to 7.00 $x10^{-6}$	0.562112
7.00 to 8.00 $x10^{-6}$	0.142492
8.00 to 9.00 $x10^{-6}$	0.029683
9.00 to 10.00 $x10^{-6}$	0.005081
10.00 to 11.00 $x10^{-6}$	0.000715
11.00 to 62.50 $x10^{-6}$	0.000091

4.3 Experiment 2 Results

To analyze the results of Experiment 2, the number of photons remaining following an attenuation were added up and averaged for a set of 1, 10, 100, and 1,000 iterations, multiplied by the energy per photon defined by Equation 3 above, and divided by the duration of the quantum signal source to derive an overall attenuated power for the given quantum signal source. Using this resulting power, an actual attenuation amount in decibels was calculated using Equation 14 below:

$$-10 \log_{10}\left(\frac{Power_{out}}{Power_{in}}\right) \tag{14}$$

where

$Power_{out}$ = *power following attenuation* (J/sec)

$Power_{in}$ = *power before attenuation* (J/sec)

To start, the power into the attenuator from the quantum signal source was defined as (581,730 photons × 1.516×10^{-19} J/photon) ÷ 400×10^{-12} seconds for an unattenuated power of 0.000220529537 J/sec. This was an average as the actual power varied from picosecond to picosecond consistent with the photon pulse profile input file used.

The full range of offset settings with dimensions set as a 25 by 25 matrix results in 25^2 or 625 unique combinations of vertical and horizontal offset pairs for every test iteration. An accompanying CD contains the full output results of Experiment 2. Table 9 below lists a subset of those results for each vertical offset with a corresponding horizontal offset setting of 0 for the four sets of test iterations run for Experiment 2:

Table 9: Experiment 2 Attenuator Result Statistics (Partial)

Lateral Offset Settings (Vertical, Horizontal)	Average of 1000 Iterations (Photons)	Average of 100 Iterations (Photons)	Average of 10 Iterations (Photons)	1 Iteration (Photons)	Difference Between the Four Sets (Photons)	Difference Between 1000 and 100 (Photons)
0, 0	0.000	0.000	0.000	0.000	0.000	0.000
1, 0	0.131	0.230	0.400	0.000	0.400	0.099
2, 0	0.896	0.700	0.100	0.000	0.896	0.196
3, 0	6.365	5.010	7.600	4.000	3.600	1.355
4, 0	40.309	38.100	41.000	29.000	12.000	2.209
5, 0	195.756	195.570	196.800	201.000	5.430	0.186
6, 0	881.787	882.710	867.800	898.000	30.200	0.923
7, 0	3553.648	3569.980	3511.500	3546.000	58.480	16.332
8, 0	11719.642	11718.180	11776.200	11656.000	120.200	1.462
9, 0	31627.353	31634.660	31576.300	31539.000	95.660	7.307
10, 0	72097.575	72099.520	72094.300	72169.000	74.700	1.945
11, 0	142716.080	142693.090	142655.900	142672.000	60.180	22.990
12, 0	239996.529	239966.500	239963.400	240027.000	63.600	30.029
13, 0	336833.855	336810.460	336889.800	336842.000	79.340	23.395
14, 0	433316.380	433330.660	433245.600	433448.000	202.400	14.280
15, 0	504753.070	504738.830	504788.800	504739.000	49.970	14.240
16, 0	546395.942	546385.230	546388.800	546299.000	96.942	10.712
17, 0	567753.809	567760.030	567714.000	567771.000	57.000	6.221
18, 0	577018.695	577024.030	577042.600	577005.000	37.600	5.335
19, 0	580337.540	580332.310	580306.400	580325.000	31.140	5.230
20, 0	581395.905	581383.360	581403.000	581395.000	19.640	12.545
21, 0	581666.074	581666.230	581669.000	581676.000	9.926	0.156
22, 0	581719.474	581721.240	581719.000	581721.000	2.240	1.766
23, 0	581728.479	581728.680	581729.500	581729.000	1.021	0.201
24, 0	581729.848	581729.880	581729.500	581730.000	0.500	0.032
25, 0	581730.000	581730.000	581730.000	581730.000	0.000	0.000

Table 9 above lists a subset of the results of Experiment 2. The first column lists the vertical and horizontal offset setting used for that row's data. Columns two through five each list the average number of photons remaining from the original quantum signal

source following attenuation for 1,000, 100, 10, and 1 iterations, respectively. Column six lists the largest difference between the four test runs and column seven lists the difference between the 1,000 and 100 test runs.

By way of comparison, the largest difference between the test runs was 239 photons occurring at vertical and horizontal offset settings of 10 and 7 between the average of 10 iterations and 1 iteration, respectively. The mode difference across all sets was 1 photon and 0 between the 1,000 and 100 averages. Likewise, the mean difference between all sets was 39 photons and 8 between the 1,000 and 100 iteration averages. By design, the same number of photons are supplied in every photon pulse, but these numbers provide evidence their distribution is consistent between photon pulses. Figures 18 through 20 below present the standard deviations for each pair of lateral offset settings for the 10, 100, and 1,000 test iteration runs:

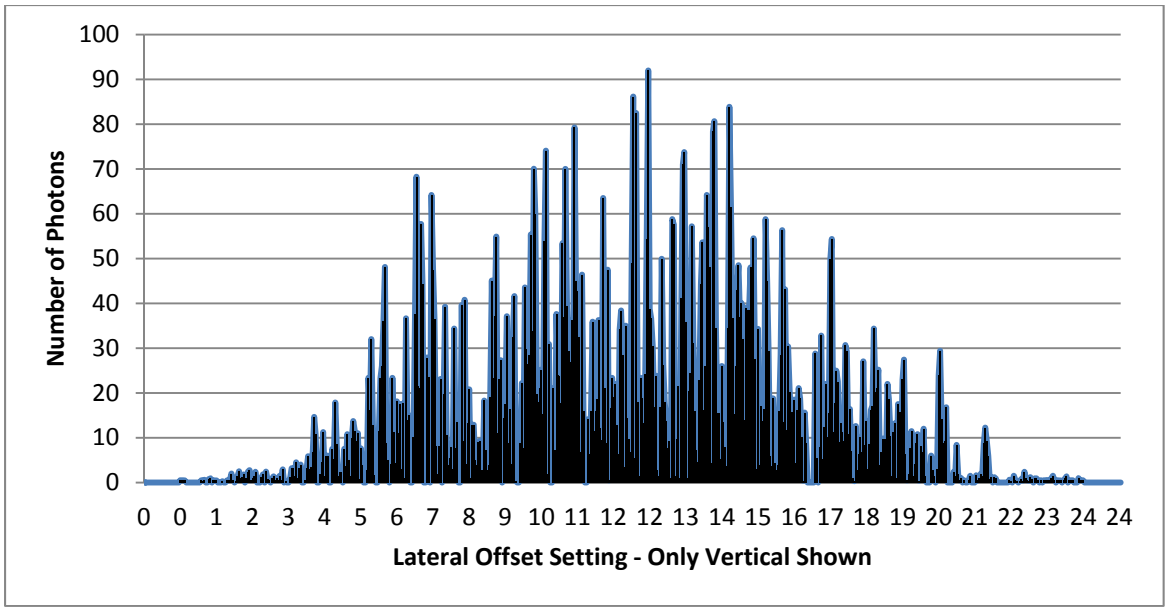

Figure 18: Standard Deviations for Lateral Offset Settings - 10 Iterations

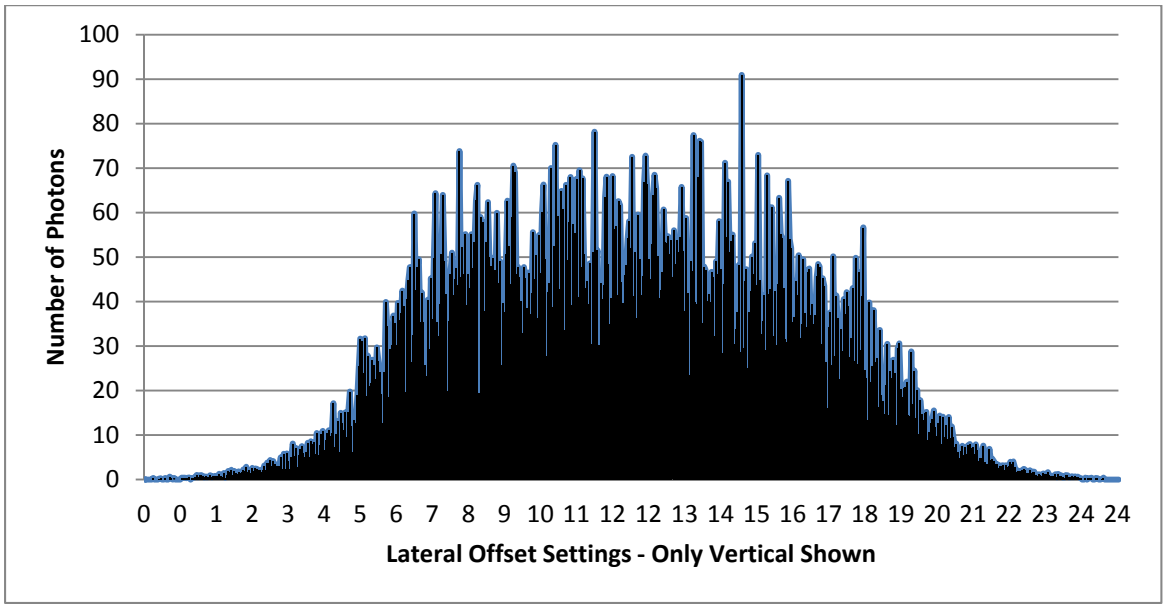

Figure 19: Standard Deviations for Lateral Offset Settings - 100 Iterations

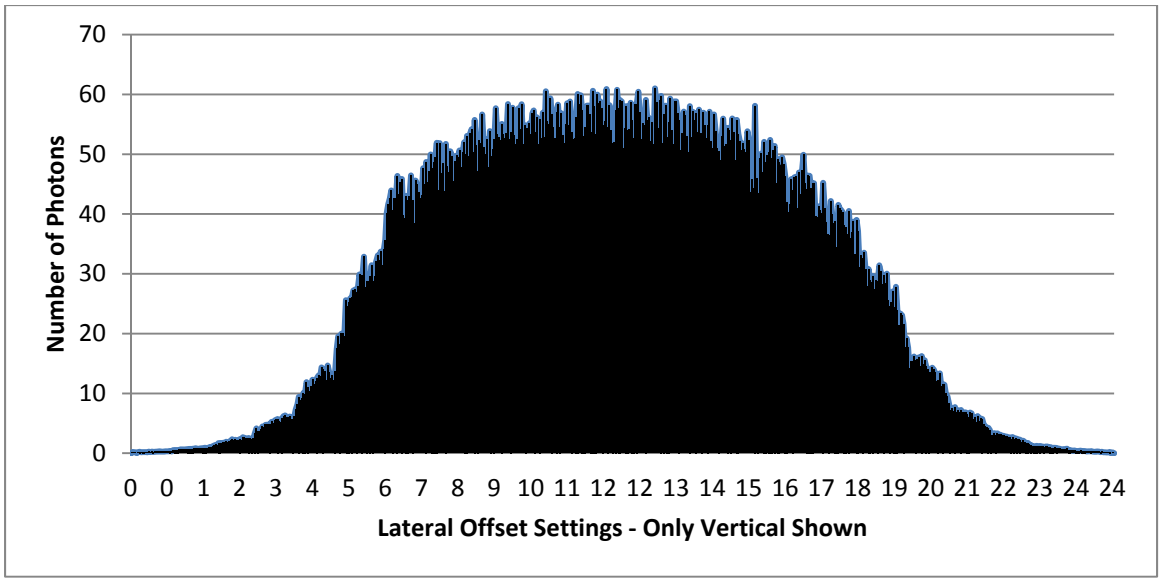

Figure 20: Standard Deviations for Lateral Offset Settings - 1000 Iterations

Figures 18 through 20 above present the standard deviations in photons for each set of lateral offset settings starting with 0, 0 and increasing to 24, 25 for the 10, 100, and 1,000 test iteration runs, respectively. As there are 25 horizontal settings between each vertical setting, only the vertical intervals are labeled on the x-axis. The figures show the standard deviation increases for lateral offset settings around the more heavily populated core regions, but never exceed roughly 90 photons between test iterations.

In order to calculate a mathematical amount of attenuation based off of Equation 11 above to compare the actual attenuation amount against, a representative value for the total amount of lateral offset is required.

From the results of Experiment 1, the photon pulse model dimensions are such that starting from the core, each region represents one micron of radial distance until the

outermost region is reached. The outermost region contains the residual percentage of photons not explicitly contained in prior regions per the Gaussian distribution. This makes its effective dimensions from the previous region's boundary to the coating which equates to a radial limit of 11 microns to 62.5 microns for Experiment 2. Because this region's dimensions are not linear as compared to all the other regions and the number of photons it contains is usually no more than one, it's in a sense an outlier and its dimensions are approximated by a comparatively small reduction to the lateral offset total of the linear regions.

Excluding the outermost region, with 11 regions defined and an outer radial limit of 11 microns, the side of an element in these regions can be calculated using Equations 12 and 13 above. The area of the circle for these regions is $\pi \cdot (11 \times 10^{-6} \text{ meters})^2$ or 3.801×10^{-10} square meters. There are a total of 23^2 or 529 elements in these regions. Dividing the circle's area by the total number of elements gives the area for a single element as $3.801 \times 10^{-10} \div 529$ or 7.1859×10^{-13} square meters. Taking the square root of this value yields the side of an individual element as 8.477×10^{-7} meters. Finally, dividing this value by the dimension of the matrix provides the value used for each lateral offset setting increment as $8.477 \times 10^{-7} \div 25$ or 3.39×10^{-8} meters.

Starting with a lateral offset of 0 meters associated with attenuator settings of vertical 24 and horizontal 25, for each decrement of the lateral offset setting, 3.39×10^{-8} meters is added to the total estimated lateral offset and a corresponding expected attenuation result is calculated using Equation 11. Table 10 below provides a subset of data with the horizontal offset at 0:

Table 10: Experiment 2 Attenuator Comparison to Expectations (Partial)

Lateral Offset Settings (Vertical, Horizontal)	Average of 1000 Iterations (Photons)	Attenuated Power of 1000 Iterations (J/sec)	Attenuator's Attenuation (dB)	Estimated Lateral Offset (m)	Calculated Attenuation (dB)
0, 0	0.000	0.000000000000	Infinite Loss	0.00002119	97.15601603
1, 0	0.131	0.000000000050	66.47450165	0.00002034	89.53898437
2, 0	0.896	0.000000000340	58.12413451	0.00001950	82.23285197
3, 0	6.365	0.000000002413	49.60923053	0.00001865	75.23761881
4, 0	40.309	0.000000015281	41.59319437	0.00001780	68.55328491
5, 0	195.756	0.000000074210	34.73006379	0.00001695	62.17985026
6, 0	881.787	0.000000334279	28.19357769	0.00001611	56.11731486
7, 0	3553.648	0.000001347162	22.14047054	0.00001526	50.36567871
8, 0	11719.642	0.000004442830	16.95807116	0.00001441	44.92494181
9, 0	31627.353	0.000011989695	12.64658615	0.00001356	39.79510416
10, 0	72097.575	0.000027331657	9.06800804	0.00001272	34.97616577
11, 0	142716.080	0.000054102610	6.10248553	0.00001187	30.46812663
12, 0	239996.529	0.000090980908	3.84516500	0.00001102	26.27098673
13, 0	336833.855	0.000127691221	2.37305725	0.00001017	22.38474609
14, 0	433316.380	0.000164267032	1.27916355	0.00000932	18.80940470
15, 0	504753.070	0.000191348153	0.61642492	0.00000848	15.54496256
16, 0	546395.942	0.000207134657	0.27213996	0.00000763	12.59141968
17, 0	567753.809	0.000215231266	0.10561405	0.00000678	9.94877604
18, 0	577018.695	0.000218743516	0.03531577	0.00000593	7.61703166
19, 0	580337.540	0.000220001666	0.01040797	0.00000509	5.59618652
20, 0	581395.905	0.000220402884	0.00249493	0.00000424	3.88624064
21, 0	581666.074	0.000220505303	0.00047727	0.00000339	2.48719401
22, 0	581719.474	0.000220525547	0.00007858	0.00000254	1.39904663
23, 0	581728.479	0.000220528960	0.00001136	0.00000170	0.62179850
24, 0	581729.848	0.000220529479	0.00000113	0.00000085	0.15544963
25, 0	581730.000	0.000220529537	0.00000000	0.00000000	0.00000000

Table 10 above lists a subset of the comparisons between the attenuation returned by the models along with the attenuations predicted by Equation 11. Column one lists the

lateral offset settings for that row's data. Column two lists the average number of photons that passed through the attenuator model for the 1,000 test iterations run and column three converts that photon count into power. Column four uses Equation 14 to convert column three's power to decibels. Column five lists the estimated lateral offset associated with column one's lateral offset settings. Finally, column six uses Equation 11 to calculate an expected attenuation amount for column five's corresponding lateral offset. An accompanying CD contains the full output results of Experiment 2.

Figure 21 below illustrates a direct comparison of the amount of attenuation resulting from the attenuator through its range of lateral offset settings with the corresponding calculated attenuation resulting from the estimated lateral offset:

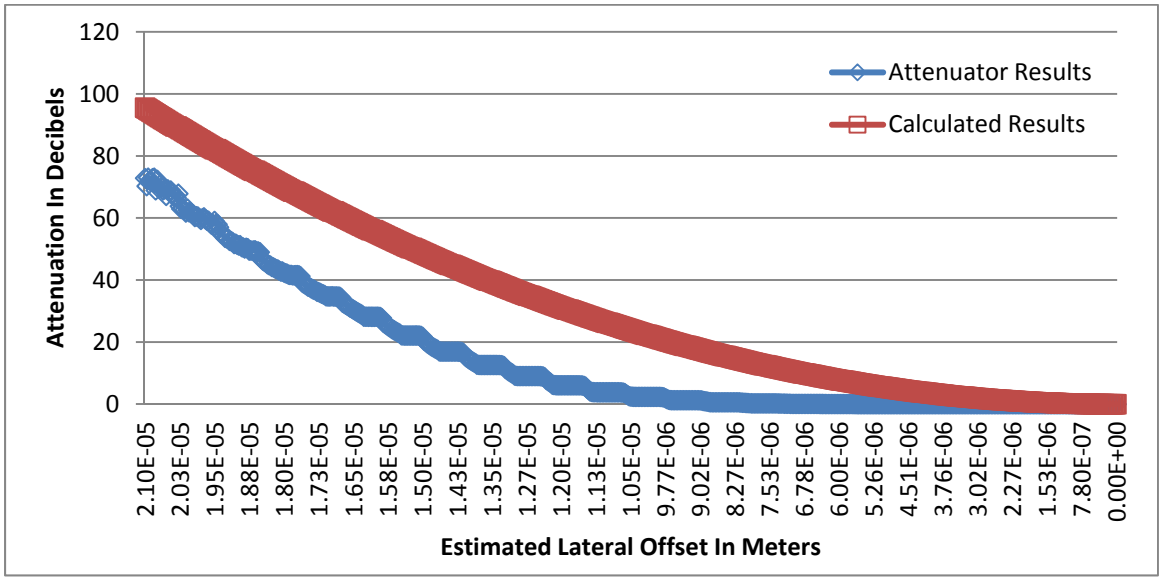

Figure 21: Experiment 2 Attenuation Direct Comparison

Figure 18 clearly shows a significant divergence between the amount of attenuation obtained from the photon pulse and attenuator models versus the expected attenuation obtained from Equation 11 above. Initially, the models' results don't attenuate as much as the mathematical result and reach infinite loss much sooner.

One possibility for the divergence in the rate of attenuation growth lies in how the attenuator model works. In the model, photons in all regions are treated equally with regard to being attenuated as the only criterion is whether an element lies within or outside the lateral offset settings. This accounts for the initially gradual rate of attenuation. The outer regions contain the fewest number of photons, so as they fall outside the lateral offset settings, we don't see much of an increase in the rate of attenuation. In contrast, the formula attenuation rate starts growing immediately and more rapidly as compared to model. This implies the attenuation due to misalignment of the core regions has a greater impact than misalignment of the cladding regions.

Another possibility for divergence between the model and the formula may lie in the approximation of the outermost region's radial range accounting for the remainder of the fiber from the outer radial limit to the coating. The typical optical fiber for single-mode operation has a cladding diameter of 125 microns outside the 8 micron core diameter. With near infinite loss not occurring with the formula until around 119.25 microns as compared to roughly 21 microns with the model, the range of the model's results appear compressed because of the non-linear outmost region.

In addition, the distribution of photons uses the Gaussian approximation for intensity for all regions instead of switching to the exponential approximation at some point within the cladding. For 1,550 nanometer light in optical fiber with a core diameter

of 9 microns, the Gaussian and exponential approximations begin to diverge around 8 microns distance from the core's center or just under double the core's radius. If 1,310 nanometer light in an optical fiber with core diameter of 8 microns experiences a similar divergence point of just prior to double the core's radius, the 8th through 11th regions of the photon pulse model would see their overall photon count percentages increase with a corresponding decrease in the photon count percentage for the outermost region. This would definitely have an effect on attenuation results due to the redistribution of photons out of the outermost region into the affected regions.

Finally, the divergence may lie in how the attenuation values are derived. In the case of the mathematical equation, the end result is strictly a function of the lateral offset for Experiment 2's conditions because the lateral offset is the only variable parameter. In contrast, the attenuation for the models is relative and based on the ratio of the output power to the input power. Given the models' results are discrete, there is a limit on the attenuation's magnitude because the numerator of Equation 14 is limited to the power supplied by a single photon while the denominator is fixed at the power supplied by the quantum signal source used for Experiment 2.

Figures 22 and 23 below compare the attenuator's results to a normalized version of the mathematical expectations over a full range of 0 meters of lateral offset and no attenuation up to the amount of lateral offset causing infinite loss:

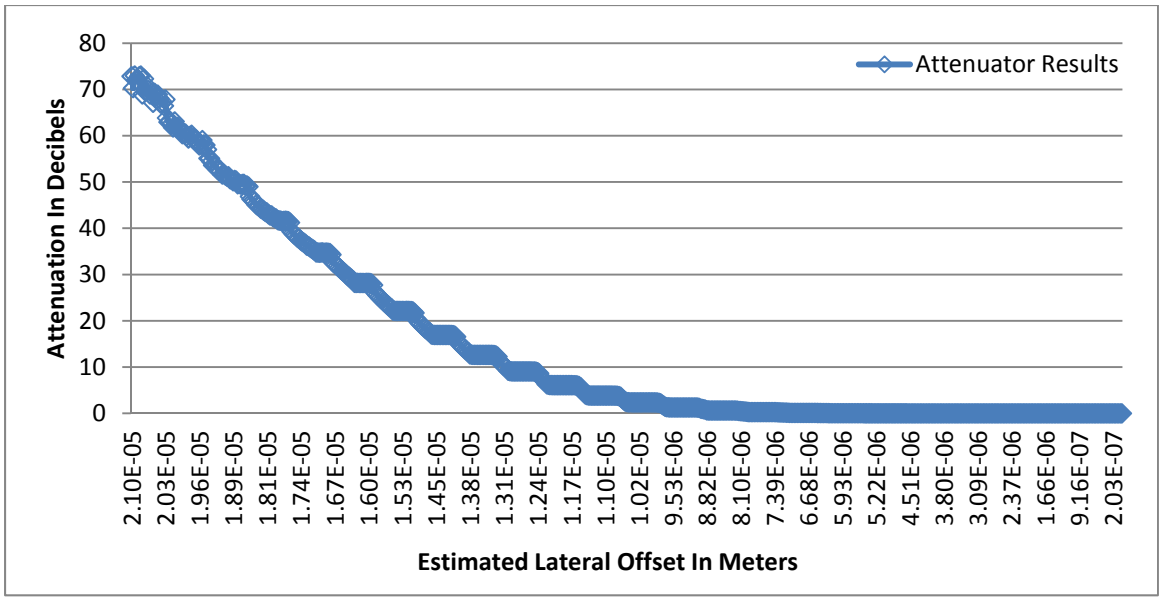

Figure 22: Experiment 2's Attenuation in dBs

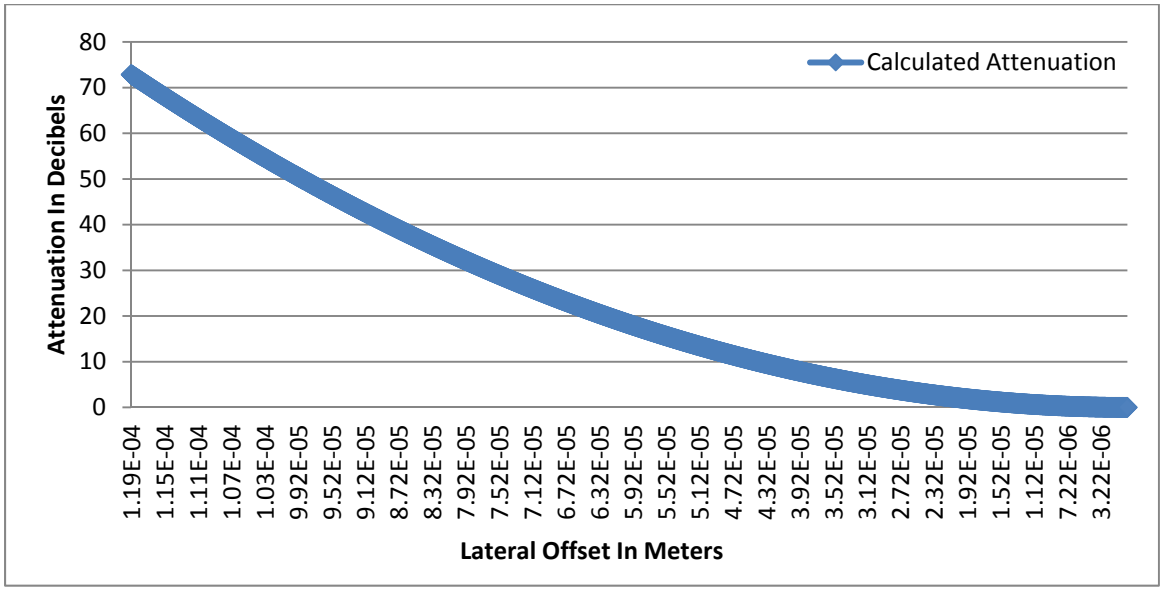

Figure 23: Mathematical Attenuation Normalized To Experiment 2's Results

While the attenuation results still diverge from the expected amounts predicted by the mathematical formula, the models provide a range of operation where the attenuator supplies the number of photons suitable for QKD simulation. That is, the models provide up to one photon with a probability dependent on lateral offset settings.

4.4 Summary

This chapter presented the results for the experiments defined in chapter 3. Experiment 1 looked at defining the spatial dimensions for modeling the quantum signal as a distribution of photons in a manner consistent with the mode field diameter such that an attenuator model using the photon pulse model would be able to return the numbers of photons expected from current QKD implementations. A current QKD quantum source was chosen and dimensions of 25 by 25 elements was determined to meet the needs identified for a successful photon pulse. Using this photon pulse model, Experiment 2 looked at supplying the attenuator model a consistent photon pulse and comparing the results of its functionality against those expected as calculated using a mathematical formula based on lateral offset. Those comparisons were found to diverge by a significant amount with possible causes attributed to differences in how parts of the models were implemented and in how the values were derived.

V. Conclusions and Recommendations

5.1 Overview

This chapter summarizes the findings drawn from analyzing the results of chapter 4, proposes the significance of the research, and recommends activities for future research related to this research's efforts.

5.2 Conclusions

This research looked at characterizing the accuracy of a QKD quantum channel attenuator using a particle based model of the quantum signal source as compared to the results expected from a mathematical formula describing the expected functionality of such an attenuator. Based off the researched characteristics of a photon pulse traveling through on optical fiber in single-mode operation, a model for representing the photon pulse as a group of individual photons placed in a three dimension model was developed and used as the source for a series of attenuation events. To add realism, the distribution of photons traveling in an optical fiber in single-mode operation was examined and determined to conform to a Gaussian distribution measured by the mode field diameter.

Using this distribution, a spatial model was developed made of concentric regions to represent individual cross-sections of an optical fiber. Using a Matlab script, a series of parameters were varied to find dimensions for the model that would be able to contain the number and distribution of photons described by the mode field diameter. In addition, the model was checked to see if the dimensions chosen would allow an optical attenuator using geometrical misalignments, specifically lateral offset misalignments, to return the numbers of photons expected from current QKD implementations.

Comparisons of the attenuation levels generated by the attenuator model and the expected attenuation levels predicted by the mathematical formula showed significant differences. It's assumed design decisions in either the photon pulse model, the attenuator model, or both caused these observed divergences.

While the attenuation levels generated do not conform to the expected levels predicted by the mathematical formula, there is a range of operation where the models developed provide the number of photons applicable for QKD simulations.

5.3 Significance of the Research

QKD implementations rely on transmitting the elementary particles of electromagnetic signals between parties securely. Since these photons can be described as both waves and particles, developers must decide whether to represent the signals as one or the other when modeling such a system. Due to the sheer number of total photons emitted by today's quantum signal sources, it would be highly resource intensive to work with individual photons starting with the quantum channel's signal source.

An alternative to this is to treat the initial quantum signal source as a single group of photons or photon pulse, reduce their total number down to the levels used in QKD implementations, and then work with this significantly reduced number of individual photons.

While this research failed to accurately model the initial components of the quantum channel as predicted by the mathematical formula defining attenuation in an optical junction, the models developed have a range of operation suitable for QKD simulation and are based on the established physical processes defining their

characteristics. The author believes the models represent a set of initial software components useful for additional development and testing for future particle-only model simulations in an overarching QKD software implementation.

In a discrete event simulation environment, one event could supply the photon pulse containing the time dimension and the attenuator could be used to schedule individual photons to remaining quantum channel components with relative times attached to each. Resolution of the time component would be down to the photon pulse's time dimension which is picoseconds for this research.

5.4 Recommendations for Future Research

This research looked at characterizing the accuracy of an optical attenuator based on an amount of lateral offset for defining its functionality using a particle-only model to represent the quantum signal source. Possible related research efforts include:

1. Modifications to the parameters defining the photon pulse model may provide insight into improved accuracy of the lateral offset attenuator's results. The consolidation of the majority of the cladding into a non-linear region may have skewed the attenuator's results. One possible avenue may involve defining the matrix's dimensions based off the mathematical formula for attenuation directly instead of dimensions based off the distribution of photons within an optical fiber cross-section. Another possibility involves retaining an overall circular model versus the square model approximation used in this research. Finally, modifications to the attenuator model to

attenuate photons based off of whether they're passing from a core region to a cladding region may improve its overall accuracy.

2. Modeling additional quantum channel components such as beam splitters and interferometers with the particle-only model to identify benefits and trade-offs of its use. Modeling of quantum channel components downstream of the attenuator may benefit from individual photons as particles versus wave approximations. A particle-only signal model alternative to wave modeling may uncover QKD simulation anomalies not immediately evident with wave models.

5.5 Summary

This chapter summarized the results of this research. After stating the overall objectives, the models for a QKD quantum signal as a representative distribution of photons in a three dimensional matrix and the lateral offset attenuator that used it were briefly outlined. The results of the experiments performed were described and the attenuator's accuracy was characterized as diverging unacceptably from expected results. While these results are inconsistent with the mathematical formula, there is a range of operation suitable for simulation in a QKD software implementation.

Next, highlights of the significance of the research included the development of an alternative quantum signal model to wave modeling of the photon pulse as well as the possibility of using the optical attenuator as the scheduler of photon events in a discrete event simulation environment.

Finally, some recommendations for future research were offered such as an improvement on the accuracy of the models developed and an expansion to other components in the quantum channel.

Appendix A: Photon Pulse and Lateral Offset Attenuator Code

```
/*****************************************************************************
 * LatOffAttenuator.cpp :  A computer program to model an optical attenuator by
 *  checking a three dimensional data structure used to represent a collection
 *  of individual photons and reducing their total amount using the concept of
 *  geometrical misalignment between opposing faces of an optical junction.
 *****************************************************************************/

#include "dSFMT.h" //Mersenne Twister pseudo random number generator.
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// The MAX_JUNCTION_DIMENSION represents the size of the sides of a square to
//  define the width and height of the face of an optical junction.  Defined to
//  be an odd value so the innermost region always has nine elements (three
//  rows of three elements).
#define MAX_JUNCTION_DIMENSION 25

// The time dimension of a three-dimensional matrix containing a photon
//  pulse.  MAX_PULSE_DEPTH represents the maximum pulse duration possible in
//  picoseconds.  With a MAX_JUNCTION_DIMENSION of 25, MAX_PULSE_DEPTH can be
//  set to a little over 1600 within the simulation environment.
#define MAX_PULSE_DEPTH 401

// Symbolic constant to set how many attenuation experiments to perform for data
//  collection.
#define NUMBER_OF_TEST_RUNS 1000


/*****************************************************************************
 * This function creates a uniformly distributed photon pulse in a three
 *  dimensional matrix by setting each element to the same value.  Useful for
 *  clearing the matrix and testing purposes.
 *****************************************************************************/
void populateUniformPhotonPulse(unsigned char bytPhotonPulse[]
                                [MAX_JUNCTION_DIMENSION]
                                [MAX_JUNCTION_DIMENSION],
                                const long lngPulseDuration,
                                const unsigned char bytNumberOfPhotons)
{
   // Loop counters for passing through the photon pulse matrix.
   unsigned short ushtDepthCounter = 0;
   unsigned short ushtVerticalCounter = 0;
   unsigned short ushtHorizontalCounter = 0;

   /**************************************************************************
    * End of the local variables and start of the function's functionality.
    */

   // Cycle through each element of the photon pulse and set it starting with
   //  the time dimension.
   for (ushtDepthCounter;
        ushtDepthCounter < lngPulseDuration;
        ushtDepthCounter++)
   {  // Reset the vertical to the top row.
      ushtVerticalCounter = 0;

      for (ushtVerticalCounter;
          ushtVerticalCounter < MAX_JUNCTION_DIMENSION;
```

```
         ushtVerticalCounter++)
      {  // Reset the horizontal for the first column on the left.
         ushtHorizontalCounter = 0;

         for (ushtHorizontalCounter;
              ushtHorizontalCounter < MAX_JUNCTION_DIMENSION;
              ushtHorizontalCounter++)
         {  // Place a constant number of photons in this element.
            bytPhotonPulse[ushtDepthCounter][ushtVerticalCounter]
                     [ushtHorizontalCounter] = bytNumberOfPhotons;
         }

      }

   }

}


/*****************************************************************************
 * This function creates a photon pulse in a three dimensional matrix where the
 *  horizontal and vertical dimensions of an optical joint have been converted
 *  into a square model approximation and segmented into discrete elements
 *  defining channels photons pass through at the optical joint.  The third
 *  dimension represents time in picoseconds.  The number of photons per
 *  picosecond is spread through concentric regions of increasing photon count
 *  in a manner consistent with the general shape of the mode field diameter as
 *  measured from the outer edges of the matrix's horizontal and vertical
 *  dimensions towards the center.  Each region represents a discrete amount of
 *  distance from the fiber's core and contains a percentage of the total photon
 *  count, currently determined in a separate Matlab script and hardcoded here.
 *  An individual photon is represented as the number one.  Using an unsigned
 *  char amount of storage, each matrix element can contain up to 255 photons.
 *****************************************************************************/
long populatePhotonPulse(unsigned char bytPhotonPulse[][MAX_JUNCTION_DIMENSION]
                [MAX_JUNCTION_DIMENSION])
{  // Index for the first dimension of the photon pulse matrix and the value
   //  returned by the function.
   unsigned short ushtCurrentPicosecond = 0;

   // Placeholder for the upper left corner of the current region and used as a
   //  fixed dimension when accessing into the photon pulse matrix.  Also used
   //  as an index into the mode field diameter profile array.
   unsigned char bytRegionCounter = 0;

   // Placeholder for the bottom right corner of the current region and used as
   //  a fixed dimension when accessing into the photon pulse matrix.
   unsigned char bytBottomRightPlaceholder = 0;

   // Index into the photon pulse matrix for the dimension that is changing.
   short shtMatrixIndex = 0;

   // Number of concentric regions defining the face of an optical junction not
   //  including the innermost region of nine elements.
   unsigned char bytNumberOfRegionsLimit = 0;

   // The number of elements in the outermost region.
   unsigned char bytElementsInOuterMostRegion = 0;

   // The number of elements in the current region (as defined by
   //  bytRegionCounter).
```

```
unsigned char bytElementsInRegion = 0;

// The total number of photons to insert in the matrix this picosecond as
//  read from the file containing the photon pulse profile.
unsigned long ulngPhotonsToInsert = 0;

// The remaining number of photons left to insert in the matrix this
//  picosecond.
unsigned long ulngPhotonsLeftToInsert = 0;

// The number of photons to place in the current region.
unsigned long ulngPhotonsForThisRegion = 0;

// Array of percentages representing each concentric regions' amount of
//  photons to contain for this picosecond, consistent with the mode field
//  diameter of an optical fiber operating in single-mode.  Determined using
//  a separate Matlab script and hardcoded here.
double dblMFDProfile[12] = {0.00000091, 0.00000715, 0.00005081, 0.00029683,
                            0.00142492, 0.00562112, 0.01822385, 0.04855931,
                            0.10635208, 0.19146176, 0.28333282, 0.34466844};

// Probability of adding a photon to the current element.
double dblProbabilityOfPhotonHere = 0.0;

// Residual probability of adding another photon to the current region.
double dblProbabilityOfOneMorePhoton = 0.0;

// File handle to read photon pulse profile.
FILE *ptrPhotonPulseProfile;

// A counter used in determing if MAX_JUNCTION_DIMENSION is odd.
short shtRemainingElements = MAX_JUNCTION_DIMENSION;

// Variable to store the state of the pseudo random number generator.
dsfmt_t PRNGState;

/*****************************************************************************
 * End of the local variables and start of the function's functionality.
 */

// Initialize the pseudo random number generator's state with a seed based
//  off the system time.
dsfmt_init_gen_rand (&PRNGState, time (NULL));

// By design, the innermost region is defined as having nine elements
//  requiring the MAX_JUNCTION_DIMENSION to be odd.  Check for this
//  condition.
while (shtRemainingElements > 1)
{  // Count up the number of regions defined by MAX_JUNCTION_DIMENSION.
   bytNumberOfRegionsLimit++;

   // Subtract two elements from the running total for each region defined
   //  by MAX_JUNCTION_DIMENSION.
   shtRemainingElements = shtRemainingElements - 2;
}

// If MAX_JUNCTION_DIMENSION is odd, set the number of elements in the
//  outermost region and remove the innermost region from the total.
if (shtRemainingElements > 0)
{  // MAX_JUNCTION_DIMENSION is odd.
   if (bytNumberOfRegionsLimit > 1)
```

```
   {  // Dimension is 5 or more. At least two regions are defined.  The
      //  outermost region is separate from the innermost region.
      //  Calculate the number of elements in the outermost region based
      //  off the dimensions of the optical junction.
      bytElementsInOuterMostRegion = (2 * MAX_JUNCTION_DIMENSION) +
                           (2 * (MAX_JUNCTION_DIMENSION - 2));

      // Decrement bytNumberOfRegionsLimit because innnermost region is
      // filled up with photons differently from other regions.
      bytNumberOfRegionsLimit--;
   }
   else
   {  // Dimension is less than 5 meaning less than 2 regions are defined.
      printf ("Fewer than two regions defined.  Exiting.\n");

      // Exit function.
      return ushtCurrentPicosecond;
   }

}
else
{  // MAX_JUNCTION_DIMENSION is even.
   printf ("Even dimensions detected.  Exiting.\n");

   // Exit function.
   return ushtCurrentPicosecond;
}

// Try to open the photon pulse profile file for reading total photon counts
//  per picosecond.
if ( (ptrPhotonPulseProfile = fopen ("LaserPulse400.csv", "r") ) == NULL)
{  // There was a problem opening the photon pulse profile file.  Notify
   //  the user.
   printf ("Failed to open the laser pulse input file.\n");
}
else
{  // File handle was successfully created.
   while (!feof (ptrPhotonPulseProfile))
   {  // Haven't reached the end of the file yet.  Get next picosecond.
      fscanf (ptrPhotonPulseProfile, "%u", &ulngPhotonsToInsert);

      // Need to keep the original total of photons to insert this
      //  picosecond read from the photon pulse profile file intact for
      //  calculating individual region photon counts based off a
      //  percentage of the original total, so use a separate variable for
      //  keeping track of the running total of photons left to insert.
      ulngPhotonsLeftToInsert = ulngPhotonsToInsert;

      // Reset the top left corner index to that of the outermost
      //  region to start filling it with this picosecond's photons.  It
      //  also serves as the index into the mode field diameter profile
      //  array.
      bytRegionCounter = 0;

      // Reset the bottom right corner index to one more than that of the
      //  outermost region to start filling it with this picosecond's
      //  photons.  It gets decrimented prior to use.
      bytBottomRightPlaceholder = MAX_JUNCTION_DIMENSION;

      // Reset the current number of elements to the number of elements
      //  in the outermost region.
```

```
bytElementsInRegion = bytElementsInOuterMostRegion;

// Cycle through each region filling it with its portion of photons
//  based off their mode field diameter profile percentage.
for (bytRegionCounter;
     bytRegionCounter < bytNumberOfRegionsLimit;
     bytRegionCounter++)
{  // Set the bounds of the bottom right corner of the current
   //  region to fill with photons.
   bytBottomRightPlaceholder--;

   // Calculate the continuous number of photons to insert in this
   //  region.
   dblProbabilityOfOneMorePhoton =
              ulngPhotonsToInsert * dblMFDProfile[bytRegionCounter];

   // Convert continuous number of photons to discrete photons by
   //  counting discrete photons until a partial photon is left.
   while (dblProbabilityOfOneMorePhoton > 1)
   {  // Count up the number of photons to insert in this region.
      ulngPhotonsForThisRegion++;

      // Decrement the running total.
      dblProbabilityOfOneMorePhoton--;
   }

   // Probabilistically determine if any partial photon becomes a
   //  discrete photon and gets added to the number of photons to
   //  insert in this region.
   if (dsfmt_genrand_close_open(&PRNGState) <
                        dblProbabilityOfOneMorePhoton)
   {  // Add another photon to put in this region.
      ulngPhotonsForThisRegion++;
   }

   // Ensure this region can contain the photons designated for it.
   if (ulngPhotonsForThisRegion > (bytElementsInRegion * 255))
   {  // There are more photons in the pulse profile than can be
      //  inserted in this region.  Notify the user and clip the
      //  pulse by inserting the maximum capacity in each element.
      printf ("Excessive photons encountered.  Clipping.\n");

      // Reset the index to traverse the column downward.
      shtMatrixIndex = bytRegionCounter;

      // Traverse region downward to the end of the column.
      for (shtMatrixIndex;
           shtMatrixIndex <= bytBottomRightPlaceholder;
           shtMatrixIndex++)
      {  // Set each element to the maximum photon count.
         bytPhotonPulse[ushtCurrentPicosecond][shtMatrixIndex]
                           [bytRegionCounter] = 255;
      }

      // Reset the index to traverse the row rightward.
      shtMatrixIndex = bytRegionCounter;

      // Traverse region rightward to the end of the row.
      for (shtMatrixIndex;
           shtMatrixIndex <= bytBottomRightPlaceholder;
           shtMatrixIndex++)
```

```
   {  // Set each element to the maximum photon count.
      bytPhotonPulse[ushtCurrentPicosecond]
                    [bytBottomRightPlaceholder]
                    [shtMatrixIndex] = 255;
   }

   // Reset the index to traverse the column upward.
   shtMatrixIndex = bytBottomRightPlaceholder;

   // Traverse region upward to the start of the column.
   for (shtMatrixIndex;
        shtMatrixIndex >= bytRegionCounter;
        shtMatrixIndex--)
   {  // Set each element to the maximum photon count.
      bytPhotonPulse[ushtCurrentPicosecond][shtMatrixIndex]
                    [bytBottomRightPlaceholder] = 255;
   }

   // Reset the index to traverse the row leftward.
   shtMatrixIndex = bytBottomRightPlaceholder;

   // Traverse region leftward to the start of the row.
   for (shtMatrixIndex;
        shtMatrixIndex >= bytRegionCounter;
        shtMatrixIndex--)
   {  // Set each element to the maximum photon count.
      bytPhotonPulse[ushtCurrentPicosecond][bytRegionCounter]
                                [shtMatrixIndex] = 255;
   }

}
else
{  // This region has space for its photons.

   // Assume mostly uniform distribution within a region by
   //  calculating the probability of inserting a photon in any
   //  one of its elements.  This only has an effect for sparse
   //  regions where the number of photons to place in the
   //  region is less than the number of elements.
   dblProbabilityOfPhotonHere = (double)
                        ulngPhotonsForThisRegion /
                        bytElementsInRegion;

   // Traverse the current region in a counterclockwise,
   //  circuitous manner until all photons have been placed.
   while (ulngPhotonsForThisRegion > 0)
   {  // Reset the index to traverse the left column downward.
      shtMatrixIndex = bytRegionCounter;

      // Traverse left side of region downward to the end of
      //  the column.
      while (ulngPhotonsForThisRegion > 0 &&
             shtMatrixIndex <= bytBottomRightPlaceholder)
      {  // Check there is room for another photon in this
         //  element.
         if (bytPhotonPulse[ushtCurrentPicosecond]
                           [shtMatrixIndex]
                           [bytRegionCounter] < 255)
         {  // Randomly determine whether to add a photon.
            if (dsfmt_genrand_close_open(&PRNGState) <
               dblProbabilityOfPhotonHere)
```

```
      {  // Add a photon to this element.
         bytPhotonPulse[ushtCurrentPicosecond]
                   [shtMatrixIndex]
                   [bytRegionCounter] =
                   bytPhotonPulse
                   [ushtCurrentPicosecond]
                   [shtMatrixIndex]
                   [bytRegionCounter] + 1;

         // Decrement the remaining number of photons
         //  to place in this region and in total.
         ulngPhotonsForThisRegion--;
         ulngPhotonsLeftToInsert--;
      }

   }

   // Increment the while counter.
   shtMatrixIndex++;
}

// Reset index to traverse the bottom row rightward.
shtMatrixIndex = bytRegionCounter;

// Traverse region rightward to the end of the bottom
//  row.
while (ulngPhotonsForThisRegion > 0 &&
       shtMatrixIndex <= bytBottomRightPlaceholder)
{  // Check there is room for another photon in this
   //  element.
   if (bytPhotonPulse[ushtCurrentPicosecond]
               [bytBottomRightPlaceholder]
               [shtMatrixIndex] < 255)
   {  // Randomly determine whether to add a photon.
      if (dsfmt_genrand_close_open(&PRNGState) <
         dblProbabilityOfPhotonHere)
      {  // Add a photon to this element.
         bytPhotonPulse[ushtCurrentPicosecond]
                   [bytBottomRightPlaceholder]
                   [shtMatrixIndex] =
                   bytPhotonPulse
                   [ushtCurrentPicosecond]
                   [bytBottomRightPlaceholder]
                   [shtMatrixIndex] + 1;

         // Decrement the remaining number of photons
         //  to place in this region and in total.
         ulngPhotonsForThisRegion--;
         ulngPhotonsLeftToInsert--;
      }

   }

   // Increment the while counter.
   shtMatrixIndex++;
}

// Reset index to traverse the right column upward.
shtMatrixIndex = bytBottomRightPlaceholder;

// Traverse region upward to the start of the column.
```

```
while (ulngPhotonsForThisRegion > 0 &&
      shtMatrixIndex >= bytRegionCounter)
{  // Check there is room for another photon in this
   //  element.
   if (bytPhotonPulse[ushtCurrentPicosecond]
               [shtMatrixIndex]
               [bytBottomRightPlaceholder] < 255)
   {  // Randomly determine whether to add a photon.
      if (dsfmt_genrand_close_open(&PRNGState) <
         dblProbabilityOfPhotonHere)
      {  // Add a photon to this element.
         bytPhotonPulse[ushtCurrentPicosecond]
                   [shtMatrixIndex]
                   [bytBottomRightPlaceholder] =
                   bytPhotonPulse
                   [ushtCurrentPicosecond]
                   [shtMatrixIndex]
                   [bytBottomRightPlaceholder]
                   + 1;

         // Decrement the remaining number of photons
         //  to place in this region and in total.
         ulngPhotonsForThisRegion--;
         ulngPhotonsLeftToInsert--;
      }

   }

   // Decrement the while counter.
   shtMatrixIndex--;
}

// Reset index to traverse the top row leftward.
shtMatrixIndex = bytBottomRightPlaceholder;

// Traverse region leftward to the start of the row.
while (ulngPhotonsForThisRegion > 0 &&
      shtMatrixIndex >= bytRegionCounter)
{  // Check there is room for another photon in this
   //  element.
   if (bytPhotonPulse[ushtCurrentPicosecond]
               [bytRegionCounter]
               [shtMatrixIndex] < 255)
   {  // Randomly determine whether to add a photon.
      if (dsfmt_genrand_close_open(&PRNGState) <
         dblProbabilityOfPhotonHere)
      {  // Add a photon to this element.
         bytPhotonPulse[ushtCurrentPicosecond]
                   [bytRegionCounter]
                   [shtMatrixIndex] =
                   bytPhotonPulse
                   [ushtCurrentPicosecond]
                   [bytRegionCounter]
                   [shtMatrixIndex] + 1;

         // Decrement the remaining number of photons
         //  to place in this region and in total.
         ulngPhotonsForThisRegion--;
         ulngPhotonsLeftToInsert--;
      }
```

```
            }

            // Decrement the while counter.
            shtMatrixIndex--;
         }

      }

   }
   // All the photons for this region have been placed.  Each
   //  region has eight fewer elements than the previous, so
   //  decrement by eight to set the number of elements in the next
   //  region.
   bytElementsInRegion = bytElementsInRegion - 8;
}

// Set the bounds of the bottom right corner of the remaining
//  region(s) to fill with photons.
bytBottomRightPlaceholder = MAX_JUNCTION_DIMENSION -
                  bytRegionCounter - 1;

// The remaining photons go in the innermost region(s), currently
//  defined as the innermost 9 elements.
if (ulngPhotonsLeftToInsert > (9 * 255))
{  // There are more photons present than can be inserted.
   //  Notify user and clip the pulse by inserting the maximum
   //  capacity in each element.
   printf ("Excessive photon count encountered.  Clipping.\n");

   do
   {
      // Reset the index to traverse the row rightward.
      shtMatrixIndex = bytNumberOfRegionsLimit;

      // Traverse region rightward to the end of the row.
      for (shtMatrixIndex;
          shtMatrixIndex <= bytBottomRightPlaceholder;
          shtMatrixIndex++)
      {  // Set each element to the maximum photon count.
         bytPhotonPulse[ushtCurrentPicosecond][bytRegionCounter]
                              [shtMatrixIndex] = 255;
      }
      // Increment to fill the next row.
      bytRegionCounter++;
   } while (bytRegionCounter <= bytBottomRightPlaceholder);

}
else
{  // The innermost region(s) can fit the remaining photons.

   // Assume mostly uniform distribution within the innermost
   //  region by calculating the probability of inserting a photon
   //  in any one of its elements.  This only has an effect for
   //  truly sparse photon pulses as the innermost region contains
   //  the highest percentage of photons and the fewest elements of
   //  any region.
   dblProbabilityOfPhotonHere = (double)
                        ulngPhotonsLeftToInsert / 9;

   // Repeatively traverse the innermost region in a top to bottom,
   //  left to right manner until all photons have been placed.
```

```
         while (ulngPhotonsLeftToInsert > 0)
         {  // There are still photons left to insert.  Reset to the
            //  first row/column of the innermost region(s).
            bytRegionCounter = bytNumberOfRegionsLimit;

            do
            {
               // Reset the index to traverse the row rightward.
               shtMatrixIndex = bytNumberOfRegionsLimit;

               // Traverse region rightward to the end of the row.
               while (ulngPhotonsLeftToInsert > 0 &&
                     shtMatrixIndex <= bytBottomRightPlaceholder)
               {  // Check there is room for another photon in this
                  //  element.
                  if (bytPhotonPulse[ushtCurrentPicosecond]
                                    [bytRegionCounter]
                                    [shtMatrixIndex] < 255)
                  {  // Randomly determine whether to add a photon.
                     if (dsfmt_genrand_close_open(&PRNGState) <
                        dblProbabilityOfPhotonHere)
                     {  // Add a photon to this element.
                        bytPhotonPulse[ushtCurrentPicosecond]
                                      [bytRegionCounter]
                                      [shtMatrixIndex] =
                                      bytPhotonPulse
                                      [ushtCurrentPicosecond]
                                      [bytRegionCounter]
                                      [shtMatrixIndex] + 1;

                        // Decrement the remaining number of photons
                        //  to place in the innermost region(s).
                        ulngPhotonsLeftToInsert--;
                     }

                  }
                  // Increment the index to fill the next element.
                  shtMatrixIndex++;
               }
               // Increment to fill the next row.
               bytRegionCounter++;

              // Continue until we reach the bottom of the region(s).
            } while (bytRegionCounter <= bytBottomRightPlaceholder);

         }

      }
      // Increment the time dimension prior to getting next picosecond
      //  from the photon pulse profile file.
      ushtCurrentPicosecond++;
   }

   // Close the photon pulse profile file.
   fclose (ptrPhotonPulseProfile);
}

// ushtCurrentPicosecond gets incremented before checking for end of file,
//  so decrement its value before returning it.  Value is actual size and
//  not an upper bound for an array.
ushtCurrentPicosecond--;
```

```
   return ushtCurrentPicosecond;
}


/*****************************************************************************
 * This function performs as a lateral offset attenuator, returning photons
 *  based off of the misalignment between opposing faces of an optical junction.
 *  The opposing faces are segmented into discrete channels with the attenuator
 *  passing on the photons whose channels line up and attenuating the others.
 *  Attenuation is defined as occurring by dropping channels from bottom up and
 *  right to left with offsets defining row/column points.  Offset adjustments
 *  are defined as starting in the upper left corner of the square model at 0,0
 *  and increasing to MAX_JUNCTION_DIMENSION.  Therefore, zero attenuation
 *  when the vertical offset setting is MAX_JUNCTION_DIMENSION or higher
 *  (including a special case when the vertical offset setting is one less than
 *  MAX_JUNCTION_DIMENSION and the horizontal offset setting is
 *  MAX_JUNCTION_DIMENSION or higher) and 100% attenuation occurs when both
 *  offsets are set to zero.  This correlates to dropping channels from bottom
 *  up and right to left.
 *****************************************************************************/
void attenuatePhotonPulse(unsigned char bytPulseToAttenuate[]
                          [MAX_JUNCTION_DIMENSION][MAX_JUNCTION_DIMENSION],
                          const long lngPulseDuration,
                          unsigned short ushtVerticalCutoff,
                          unsigned short ushtHorizontalCutoff)
{
   // Loop counters for indexing each of the photon pulse matrix's three
   //  dimensions.
   unsigned short ushtDepthCounter = 0;
   unsigned short ushtVerticalCounter = 0;
   unsigned short ushtHorizontalCounter = 0;

   /***************************************************************************
    * End of the local variables and start of the function's functionality.
    */

   // Check for the special condition where horizontal offset is equal to or
   //  greater than MAX_JUNCTION_DIMENSION indicating no attenuation of the row
   //  indexed by the vertical offset setting.  This combination of offset
   //  settings is duplicated by fully attenuating the next row.
   if (ushtHorizontalCutoff >= MAX_JUNCTION_DIMENSION)
   {  // Horizontal offset indicates no attenuation of this row.  Adjust
      //  vertical offset to next row...
      ushtVerticalCutoff++;
      //  and reset the horizontal offset to start at the beginning of the
      //  row.
      ushtHorizontalCutoff = 0;
   }

   // Check for zero attenuation condition to skip attenuating.
   if (ushtVerticalCutoff < MAX_JUNCTION_DIMENSION)
   {  // Cycle through each segment of the optical junction outside the offset
      //  bounds setting all elements to zero i.e. attenuated.  Those within
      //  the bounds are passed on as is.
      for (ushtDepthCounter;
           ushtDepthCounter < lngPulseDuration;
           ushtDepthCounter++)
      {  // Reset the vertical and horizontal starting point for this
         //  picosecond.
         ushtVerticalCounter = ushtVerticalCutoff;
```

```
         ushtHorizontalCounter = ushtHorizontalCutoff;

         for (ushtVerticalCounter;
              ushtVerticalCounter < MAX_JUNCTION_DIMENSION;
              ushtVerticalCounter++)
         {

            for (ushtHorizontalCounter;
                 ushtHorizontalCounter < MAX_JUNCTION_DIMENSION;
                 ushtHorizontalCounter++)
            {  // Any photons contained in this element were attenuated.
               bytPulseToAttenuate[ushtDepthCounter][ushtVerticalCounter]
                                              [ushtHorizontalCounter] = 0;
            }

            // Reset the horizontal for attenuating the next row.  Don't
            //  place before the loop or the initial horizontal offset
            //  setting will have no affect.
            ushtHorizontalCounter = 0;
         }

      }

   }

}


/*****************************************************************************
 * This function provides information on the results of the attenuator's
 *  affects on the photon pulse.
 *****************************************************************************/
unsigned long checkOutput(unsigned char bytAttenuatedPhotonPulse[]
                          [MAX_JUNCTION_DIMENSION][MAX_JUNCTION_DIMENSION],
                          const long lngPulseDuration)
{
   // Variable to store the number of photons remaining in the attenuated
   //  pulse and returned by the function.
   unsigned long ulngNumberOfRemainingPhotons = 0;

   // Loop counters for indexing each of the photon pulse matrix's three
   //  dimensions.
   unsigned short ushtDepthCounter = 0;
   unsigned short ushtVerticalCounter = 0;
   unsigned short ushtHorizontalCounter = 0;

   /**************************************************************************
    * End of the local variables and start of the function's functionality.
    */

   // Cycle through each element of the photon pulse within the pulse duration
   //  and count the number of photons left.
   for (ushtDepthCounter;
        ushtDepthCounter < lngPulseDuration;
        ushtDepthCounter++)
   {  // Reset the vertical to start counting at the first row.
      ushtVerticalCounter = 0;

      for (ushtVerticalCounter;
           ushtVerticalCounter < MAX_JUNCTION_DIMENSION;
          ushtVerticalCounter++)
```

```
      {  // Reset the horizontal to start counting from the first element of
         //  this row.
         ushtHorizontalCounter = 0;

         for (ushtHorizontalCounter;
              ushtHorizontalCounter < MAX_JUNCTION_DIMENSION;
              ushtHorizontalCounter++)
         {  // Add the photons in this matrix element to the running total.
            ulngNumberOfRemainingPhotons = ulngNumberOfRemainingPhotons +
                                    bytAttenuatedPhotonPulse[ushtDepthCounter]
                                    [ushtVerticalCounter]
                                    [ushtHorizontalCounter];
         }

      }

   }
   // Return the running total of photons remaining after attenuation.
   return ulngNumberOfRemainingPhotons;
}


/*****************************************************************************
 * This function writes model performance statistics to a file.
 *****************************************************************************/
void recordStatistics (const unsigned short ushtVerticalOffsetUsed,
                       const unsigned short ushtHorizontalOffsetUsed,
                       const unsigned short ushtTestIteration,
                       const unsigned long ulngPhotonsAfterAttenuation)
{
   // File handle to record output of model performance.
   FILE *ptrResults;

   /*****************************************************************************
    * End of the local variables and start of the function's functionality.
    */

   // Try to open the performance output file for appending.
   if ( (ptrResults = fopen ("Lateral_Offset_Attenuator_Output.csv", "a") )
      != NULL)
   {  // File handle was successfully created.  Append the file with the
      //  performance parameters supplied.
      fprintf (ptrResults, "%u,%u,%u,%u\n", ushtVerticalOffsetUsed,
                       ushtHorizontalOffsetUsed, ushtTestIteration,
                                    ulngPhotonsAfterAttenuation);

      // Close the performance output file.
      fclose (ptrResults);

      // Provide feedback to user where the program is at.
      printf ("%u, %u, %u : ", ushtVerticalOffsetUsed,
                            ushtHorizontalOffsetUsed, ushtTestIteration);
   }
   else
   {  // There was a problem opening the performance output file.  Notify the
      //  user.
      printf ("\nFailed to open the output file.\n");
   }

}
```

```
/*****************************************************************************
 * The main entry point for this program.
 *****************************************************************************/
void main()
{  // A three dimensional matrix to represent a photon pulse.
   unsigned char bytArrayPhotonPulse[MAX_PULSE_DEPTH][MAX_JUNCTION_DIMENSION]
                         [MAX_JUNCTION_DIMENSION] = {0};

   // Loop counters to run increasing attenuator offset settings.  Increasing
   //  attenuator offset settings equate to reduced attenuation.  Setting
   //  ushtVerticalOffset offset setting to MAX_PULSE_DEPTH or more results in
   //  no attenuation i.e no geometrical misalignment.  Setting
   //  ushtHorizontalOffset to MAX_PULSE_DEPTH or more results in no
   //  attenuation of the row indexed by ushtVerticalOffset and is equivalent
   //  to fully attenuating the next row.
   unsigned short ushtVerticalOffset = 0;
   unsigned short ushtHorizontalOffset = 0;

   // Loop counter for multiple test iterations for a pair of offset settings.
   unsigned short ushtTestRunCounter = 0;

   // Variable to store the size of the photon matrix in the time dimension.
   //  Change the initialization value for functionality testing.
   long lngPulseDuration = 0;

   // Variable to store the number of photons to insert in a uniformly
   //  distributed photon pulse matrix.  Normally set to zero to clear the
   //  matrix between attenuation test iterations of offset pairs, but also
   //  used for functionality testing purposes.
   unsigned char bytUniformPhotonCount = 0;

   // Variable to store the number of photons in a pulse following an
   //  attenuation test for a pair of offset settings.
   unsigned long ulngRemainingPhotons = 0;

   /***************************************************************************
    * End of the local variables and start of the function's functionality.
    */

   // A loop to performing multiple attenuation events while increasing the
   //  amount of vertical offset designated.  This equates to moving from
   //  maximum attenuation (maximum lateral offset misalignment) to zero
   //  attenuation (no lateral offset misalignment).
   for (ushtVerticalOffset;
        ushtVerticalOffset < MAX_JUNCTION_DIMENSION;
        ushtVerticalOffset++)
   {  // Reset the horizontal offset setting for this row.
      ushtHorizontalOffset = 0;

      // Another loop to performing multiple attenuation events while
      //  increasing the amount of horizontal offset designated.
      for (ushtHorizontalOffset;
           ushtHorizontalOffset <= MAX_JUNCTION_DIMENSION;
           ushtHorizontalOffset++)
      {  // Reset the loop counter for multiple test iterations for this pair
         //  of offset settings.
         ushtTestRunCounter = 0;

         // A loop to perform multiple test runs with the same pair of offset
```

```
         //  settings.
         for (ushtTestRunCounter;
              ushtTestRunCounter < NUMBER_OF_TEST_RUNS;
              ushtTestRunCounter++)
         {  // Clear the photon pulse matrix from the last test.  Also used
            //  for creating uniformly distributed photon pulses for
            //  functionality testing purposes by commenting out
            //  populatePhotonPulse() below.
            populateUniformPhotonPulse(bytArrayPhotonPulse,
                                       lngPulseDuration + 1,
                                       bytUniformPhotonCount);

            // Populate three dimensional matrix from file.  Comment out to
            //  perform functionality testing using
            //  populateUniformPhotonPulse() above.
            lngPulseDuration = populatePhotonPulse(bytArrayPhotonPulse);

            // Attenuate the number of photons based off the lateral offset
            //  settings.  Increasing settings equate to decreasing lateral
            //  misalignment and corresponding decrease in attenuation.
            attenuatePhotonPulse(bytArrayPhotonPulse, lngPulseDuration,
                                 ushtVerticalOffset, ushtHorizontalOffset);

            // Check the output of the attenuation function.
            ulngRemainingPhotons = checkOutput(bytArrayPhotonPulse,
                                               lngPulseDuration);

            // Record performance results in a file.
            recordStatistics(ushtVerticalOffset, ushtHorizontalOffset,
                             ushtTestRunCounter, ulngRemainingPhotons);

         }

      }

   }

   // A prompt and a pause to allow collecting memory usage statistics.
   printf ("\n\nDone.  Press enter.\n");
   getchar();
}
```

Appendix B: Matlab Script For Experiment 1

```
numberOfRegions = 11;

outerRadialLimit = 11*10^-6;

interval = outerRadialLimit / numberOfRegions;

radialDistance = 0:interval:outerRadialLimit;

wavelength = 1310*10^-9;

radiusOfCore = 4*10^-6;

coreRefractiveIndex = 1.49;
cladRefractiveIndex = 1.485;

numericalAperture = sqrt(coreRefractiveIndex^2 - ...
                         cladRefractiveIndex^2);

normalizedFrequency = (2 * pi * radiusOfCore * numericalAperture) / ...
                                                wavelength;

rightside = 0.65 + 1.62 * normalizedFrequency^(-3/2) ...
                 + 2.88 * normalizedFrequency^-6;

modefieldradius = rightside * radiusOfCore;

intensity = 1 / modefieldradius^2 * exp( (-2 * radialDistance.^2) / ...
                                                modefieldradius^2);

sigma = modefieldradius / 2;

CDF = erf(radialDistance / sqrt(2 * sigma^2));

regionTotal = 0;

for counter = 1:length(radialDistance) - 1
    fprintf('Percent from %02.2f to %02.2f microns = %f%%\n', ...
            radialDistance(counter) / 10^-6, ...
            radialDistance(counter + 1) / 10^-6, ...
            100 * (CDF(counter + 1) - CDF(counter)))

    regionTotal = regionTotal + 100 * (CDF(counter+ 1) - CDF(counter));
end


fprintf('Region total is %f%%\n', regionTotal)
```

Bibliography

[1] J. Walker, Fundamentals of Physics 8th Edition, Hoboken: John Wiley & Sons, Inc, 2008.

[2] B. Lehnert, "Joint Wave-Particle Properties of the Individual Photon," *Progress In Physics*, vol. 4, pp. 104 - 108, 2007.

[3] A. Oliviero and B. Woodward, Cabling The Complete Guide to Copper and Fiber-Optic Networking 4th Edition, Indianapolis: Wiley Publishing, Inc., 2009.

[4] J. M. Senior and M. Y. Jamro, Optical Fiber Communications Principles and Practice 3rd Edition, Harlow: Pearson Education Limited, 2009.

[5] H.-J. Hagemann and D. U. Wiechert, "Measurements and Calculations of the LP01 Intensity of SM Fibers Far Off the Core," *Journal of Lightwave Technology,* vol. 10, no. 4, pp. 407 - 412, 1992.

[6] B.D. Guenther, A. Miller, L. Byvel, J. E. Midwinter, Encyclopedia of Modern Optics, Academic Press, 2004.

[7] W. Trappe and L. C. Washington, Introduction to Cryptography with Coding Theory, Pearson Education, Inc., 2006.

[8] S. Wiesner, "Conjugate Coding," *ACM SIGACT News - A special issue on cryptography,* pp. 78 - 88, 1983.

[9] R. Bose, Information Theory, Coding and Cryptography, New Delhi: Tata McGraw Hill Publishing Company Limited, 2008.

[10] Department of Defense, DoD Directive 5000.59: Washington: GPO, 2007.

[11] M. Born and E. Wolf, Principles of Optics 7th Edition, Cambridge: Cambridge University Press, 1999.

[12] C. H. Bennett and G. Brassard, IEEE, Ed., Bangalore, 1984.

[13] I. R. Kenyon, The Light Fantastic A Modern Introduction to Classical and Quantum Optics, New York: Oxford University Press, 2008.

[14] Owens Corning, "Mode-Field Diameter Measurement Method," August 2001. [Online]. Available: http://www.corning.com/docs/opticalfiber/mm16_08-01.pdf. [Accessed 29 January 2012].

[15] N. Gisin, G. Ribordy, W. Tittel and H. Zbinden, "Quantum Cryptography," *Reviews of Modern Physics,* vol. 74, no. 1, pp. 145 - 195, 2002.

[16] C. P. Williams, Explorations In Quantum Computing, 2nd Edition, London: Springer-Verlag London Limited, 2011.

[17] M. Saito and M. Matsumoto, "SIMD-Oriented Fast Mersenne Twister," 18 April 2009. [Online]. Available: http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/SFMT/index.html. [Accessed 14 February 2012].

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
22-03-2012	Master's Thesis	Aug 2010 - Mar 2012

4. TITLE AND SUBTITLE
Empirical Analysis Of Optical Attenuator Performance In Quantum Key Distribution Systems Using a Particle Model

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Adams, Thomas C., Captain, USAF

5d. PROJECT NUMBER
11V201

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/ENG)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GCS/ENG/12-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Dr. Gerald Baumgartner
Laboratory for Telecommunications Sciences
8080 Greenmead Drive
College Park, MD 20740
(240) 373-2743
gbbaumg@gmail.com

10. SPONSOR/MONITOR'S ACRONYM(S)
LTS

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED

13. SUPPLEMENTARY NOTES
This material is declared a work of the United States Government and is not subject to copyright protection in the United States.

14. ABSTRACT
Quantum key distribution networks currently represent an active area of development and software modeling to address the security of future communications. One of the components used in quantum key distribution implementations is an optical attenuator. Its role in the system is necessary to reach the single photon per bit necessary to maintain theoretically perfect secrecy. How the photon pulse is modeled has a significant impact on the accuracy and performance of quantum channel components like the optical attenuator. Classical physics describe light using Maxwell's wave equations for electromagnetism. Quantum physics has demonstrated light also behaves as discrete particles referred to as photons. This paper looks at characterizing the accuracy of a software model of an optical attenuator as might be used in a quantum key distribution system using a particle-only model of the photon pulse.

15. SUBJECT TERMS
Model, Simulation, QKD, Attenuator, Photon

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Jeffrey W. Humphries, Lt Col, USAF
U	U	U	UU	101	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-6565, x 7253 (jeffrey.humphries@afit.edu)

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18